commodore

VIC-20

Személyi számítógép

Felhasználói kézikönyv

VIC-20

Személyi számítógép

FELHASZNÁLÓI KÉZIKÖNYV

Az eredeti kiadás címe:
Personal computing on the VIC 20
A friendly computer guide
COMMODORE INTERNATIONAL, LTD
Computer System Division
Norristown, PA 19403
U.S.A.

© 1981 Commodore International and Avalanche Productions
Minden jog fenntartva.

A kiadványt az 1969. évi III. törvény 13. § és a 18. § első bekezdése alapján tilos bármilyen módon sokszorosítani és a sokszorosítást terjeszteni.

A kiadásért felel a NOVOTRADE RT. igazgatója
Készült az ÁRKUS GM gondozásában.

ELŐSZÓ

Amennyiben ezek az Ön első lépései a számítógép felé, ez a kézikönyv kiváló bevezetést nyújt a számítástechnikába. Ellentétben a többi hasonló kiadvánnyal, ezt nem kell végigolvasnia, hogy a kívánt információkat megszerezze. Az első fejezet (A kezdet) elolvasása után közvetlenül az Önt érdeklő fejezetre ugorhat. Amennyiben a rajzolás érdekli, forduljon a 4. fejezethez, amennyiben a zene, próbálkozzon az ötödikkel.
Minden fejezet első oldalán található az induláshoz egy egyszerű program. Gépelje be a programot pontosan úgy, ahogy látja, és nézze meg, mi történik. A fejezet további részeiben elmagyarázzák, mit is tett, és mit lehet még tenni. A 7. fejezet összefoglalja a legfontosabb programozási elveket, és elmagyarázza a mintaprogramokban alkalmazott programozási technikákat.
A gyakorlott programozó a VIC-et mikroszámítógépként használja. A Commodore gépekhez való hasonlatossága további segítséget nyújt, mivel a BASIC-je és a grafikája közel azonos a PET/CBM-vel. További hivatkozási anyag, programozási információk a mellékletekben találhatók.
A VIC-hez rendelkezésre állnak perifériák, tartozékok: a VIC kazettás magnetofon, hajlékony mágneslemez-egység, telefon modem, nyomtatók, hogy csak néhányat említsünk (ld. A. Melléklet). A számítógépek egyre nagyobb szerepet játszanak mindennapi életünkben — akár otthon, az iskolában, az üzleti életben. A VIC nemcsak bevezeti önt a számítógépek világába, de a lehetőséget is megadja, hogy bővítse azt a világot.

TARTALOMJEGYZÉK

ELŐSZÓ 3

1. MEGISMERKEDUNK A VIC-KEL 7
   A kezdet 8
   Az ön első számítógépes programja 10

2. KÉPERNYŐ ÉS BILLENTYUZET HASZNÁLATA 14
   Az ön első grafikus karaktere 15
   Ismerkedés a VIC 20 billentyűzettel 16
   A VIC 20 kalkulátor 21
   Bevezetés a színekbe 22

3. SZIN ÉS GRAFIKÁK 24
   Programozás színesben 24
   A VIC színbillentyűk 26
   Az alap és háttér színeinek változtatása 27
   Képernyő-színezés 29
   Véletlen színek 30
   Billentyűzet grafikája 34
   Címek, fejlécek grafikája 35

4. ANIMÁCIÓ 37
   Repülő madarak 38
   Ugráló labda 42
   Kurzor ellenőrzés 43
   Életrekeltés POKE-kal és PEEK-kel 44

5. HANG ÉS ZENE 49
   Hangképzés 50
   A VIC négy hangja 51
   Fehér zaj generátor 53
   VIC mint zongora 54
   Dalok lejátszása 56
   Néhány szó a POKE-ról 58

6. BESZÉLGETÉS A VIC-KEL 59
   Változók bevezetése 62
   Válasszon egy hangot 63
   A GET utasítás 64

7. BEVEZETÉS A PROGRAMOZÁSBA 69
   Az ön első BASIC utasításai 70
   Véletlen számok 76

MELLÉKLETEK 79
A) VIC rendszer tartozékok 80
B) Kazettás magnó használata 82
C) VIC BASIC szótár 85
D) BASIC utasítás rövidítések 103
E) Aláp — háttér színkombinációk 105
F) Zenei hangok 106
G) Egyszerű hanghatások 106
H) Képernyőkódok 110
I) Képernyő memória térkép 114
J) ASCII és Karakter ($) kódok 115
K) Származtatott matematikai függvények 118
L) INPUT/OUTPUT készülékek lábkiosztása 119
M) Kipróbálásra ajánlott VIC-programok 123
N) Hibajelzések 125

1. FEJEZET

MEGISMERKEDÜNK A VIC-KEL

Végre kezdhetjük

Az ön első számítógépes programja

Próbálja meg begépelni ezt a programot pontosan úgy, ahogy itt látja, és nézze meg, mi történik.

1 PRINT "VIC 20" és nyomja meg a RETURN gombot
2 GOTO 1 és nyomja meg a RETURN gombot
RUN és nyomja meg a RETURN gombot.

A program megállításához nyomja meg a RUN/STOP gombot.

Megjegyzések a programhoz:

1. sor. Közli a VIC-kel, hogy írja ki, ami az idézőjelek között van.
2. sor. Közli a VIC-kel, hogy ugorjon az 1. sorra, és írja ki azt újra.
3. sor. A RUN szó begépelése hatására kezd el futni a program.

A KEZDET – GYAKOROLJUNK EGY KICSIT

Megcsinálta. A VIC képernyője most gyönyörű színekben pompázik, és várja, hogy mit tegyen. A sötétkék villogó négyszög, melyet kurzornak neveznek, egy jelzés a VIC-től, hogy önre vár.

```
. . .  CBM BASIC V2  . . .

35 83 BYTES FREE

READY
```

VIC AJÁNLÁS

Ha egy olyan karaktert ütött be, amit nem akart, nyomja meg az INST/DEL billentyűt. Ekkor eltűnik a közvetlenül a villogó kurzor előtt lévő karakter. Annyiszor használja ezt a billentyűt, ahány nem kívánt karaktert meg akar szüntetni.

Akkor lássunk hozzá! Kezdje a következő billentyűk lenyomásával:

PRINT

Figyelje meg, hogyan mozog minden alkalommal egy pozíciót a kurzor, amikor megnyomja a billentyűt. A kurzor megmutatja önnek, hová kerül a következő karakter a képernyőn. Most keresse meg a SHIFT gombot. Kettő van belőle,egyformák.

Nyomja le a SHIFT billentyűt, és mialatt ezt lenyomva tartja, nyomja le: a [" 2] gombot.

Miután megnyomta, elengedheti a SHIFT gombot.

A képernyő a következőképp néz ki:

```
. . .  CBM BASIC V2  . . .

3583 BYTES FREE

READY

PRINT"
```

A [" 2] gomb lenyomása, miközben lenyomva tartotta a SHIFT gombot, azt eredményezte, hogy az idézőjel megjelent a képernyőn. Most nyomja meg a következő billentyűket:

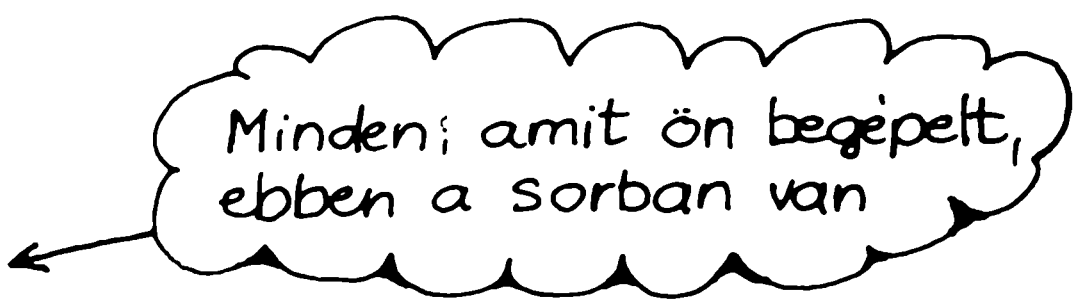
SZIVÁRVÁNY

Végül nyomja le a SHIFT billentyűt, és nyomja meg újra a [" 2]-t. A képernyő a következőképp fog kinézni:

... CBM BASIC V2 ...

READY

PRINT "SZIVÁRVÁNY"



Amennyiben a 2-es szám jelenik meg a képernyőn idézőjel helyett, nem nyomta le a SHIFT billentyűt. Nyomja le az INST/DEL billentyűt, így kitörli a 2-t, és próbálja meg újra.
Keresse meg a RETURN billentyűt.
Nyomja meg a RETURN billentyűt, és nézze meg a képernyőt.

... CBM BASIC V2 ...

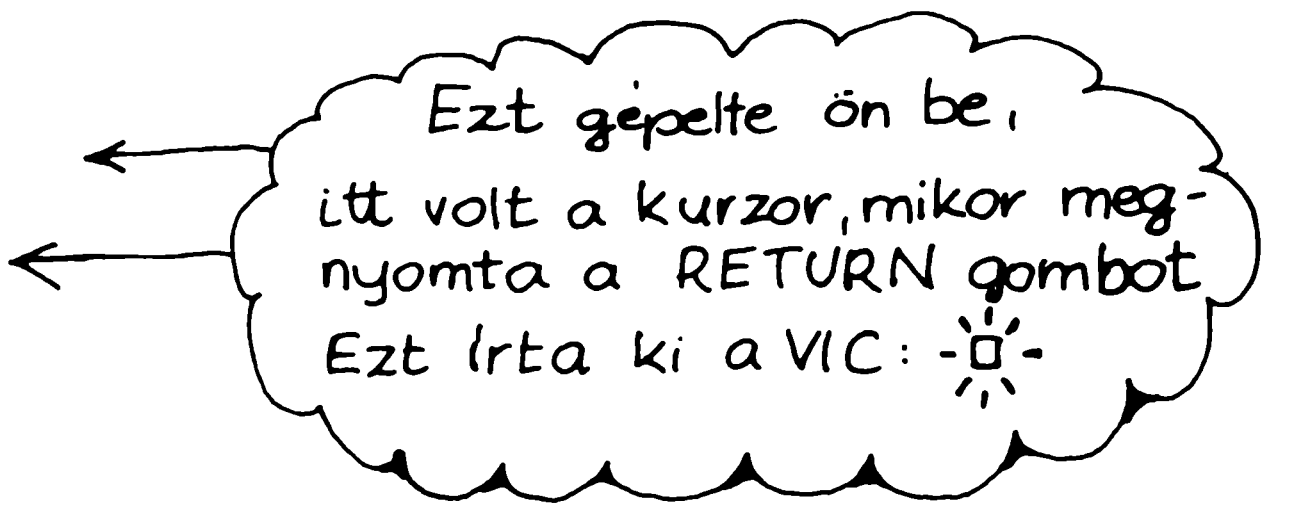
3583 BYTES FREE

READY

PRINT "SZIVÁRVÁNY"

SZIVÁRVÁNY

READY



A RETURN billentyű lenyomásával közölte a VIC-kel, hogy befejezte a gépelést. A VIC ekkor megnézte, amit ön begépelt, és felismerte, hogy van tennivalója. (Jelen esetben az, hogy írjon ki valamit.)

Ezután a VIC kiír mindent, ami az idézőjelek között volt.

Miután a VIC befejezte a SZIVÁRVÁNY kiírását, ezt közli önnel, úgy, hogy kiírja a READY (kész) üzenetet, és villog a kurzor.

Ön következik. Üssön be néhány PRINT üzenetet, melyet a VIC kiírjon a képernyőre. Próbálja egyedül a következőket:

PRINT "JÓ NAPOT"

(Ne felejtse el, hogy az idézőjelhez le kell nyomni a SHIFT gombot.)

Nyomja meg a RETURN-t

PRINT "HELLÓ"

és RETURN gomb

PRINT "KATI"

és újra RETURN gomb.

Gyakorolja a betűkön kívüli karakterek használatát is az idézőjelek között. Ezt a VIC nem bánja. *Megjegyzés:* Amennyiben elgépeli a PRINT szót, ezt közli a VIC önnel úgy, hogy a következő üzenetet írja ki a képernyőre.

```
? SYNTAX
  ERROR
```

Ne izguljon. Semmilyen módon nem lehet kárt tenni a VIC-ben a billentyűzetről. Ha hibázik, a VIC felhívja a figyelmét a hibára. Az N) Mellékletben megtalálja a hibaüzeneteket és azok jelentését. Itt most ne törődjön a SYNTAX ERROR (szintaktikai hiba) üzenettel. Folytassa a gyakorlást.
Rövid időn belül a képernyő megtelik mindazzal, amit begépelt. A VIC kiválóan tudja törölni ezt a halmazt. Ahhoz, hogy a VIC törölje a képernyőt, tegye a következőt:
Tartsa lenyomva a SHIFT gombot, és nyomja meg a CLR/HOME billentyűt.
A képernyő azonnal tiszta lesz. Mindent, amit ön és a VIC írt, eltűnik. Csak egy tiszta, fehér képernyőt lát, kék, villogó kurzorral a bal felső sarokban. Tartsa emlékezetében ezt a hőstettet. A képernyő kitisztítása az egyik leggyakrabban használatos utasítás.

AZ ÖN ELSŐ SZÁMÍTÓGÉPES PROGRAMJA

Ha már a VIC képes kiírni a képernyőre az ön üzeneteit, lépjünk tovább. Kezdjük szerkeszteni az ön első számítógépes programját.

Első lépés: Töröljük a képernyőt úgy, hogy tartsuk lenyomva a SHIFT gombot, és üssük le a CLR/HOME billentyűt.

Második lépés: Gépeljük be: NEW és nyomjuk meg a RETURN gombot.

Harmadik lépés: Gépeljük be: 1 PRINT " [SPACE] VIC 20";

(Megjegyzés: az idézőjel és pontosvessző esetén használja a SHIFT gombot. A SPACE a billentyűzet alsó részén levő hosszú gomb.)

Negyedik lépés: Gépelje be: 2 GOTO 1 és utána nyomja be a RETURN billentyűt.

Az üres helyeket mindig ezzel a hosszú nyomógombbal képezhetjük.

Miután elkészült, a képernyő a következőképp néz ki:

```
NEW

READY
1     PRINT"    VIC 20"
2     GOTO 1
```

A kurzor villog, ami azt jelenti, hogy a VIC vár.
Ha minden rendben levőnek tűnik, gépelje be a következő szót:

RUN

és utána nyomja meg a RETURN gombot.

A képernyő tele lesz VIC 20 felirattal. Úgy tűnik, mintha a képernyő élő betűkkel lenne tele, melyek futnak a képernyőn.

VIC AJÁNLÁS: A programhibák javítása

Ha a programsorban hibát követ el, a következő javítási, szerkesztési lehetőségek vannak:

1. Újragépelheti a sort, és a VIC automatikusan helyettesíti a régit az újjal. Például, ha a program a következőképp néz ki:

```
10    PRINT "VIC 20"
20    GOTO 10
```

A RETURN többszöri lenyomásával ugorjon lejjebb a kurzorral a képernyőn, majd gépelje be:

```
10    PRINT "VIC 20"
```

majd nyomja le a RETURN-t.

Most az új sor került a régi helyére, és a program futásképes. Azért, hogy biztos legyen a dolgában, gépelje be: LIST és nyomja be a RETURN billentyűt.
A sorok cseréje, helyettesítése gyors és könnyű javítási módszer.

2. A nem kívánt sort kitörölheti úgy, hogy begépeli a sor számát, és lenyomja a RETURN-t. Igy a teljes sort kitörli a memóriából.

3. A sor javítása úgy is megtörténhet, hogy a kurzort az adott karakterre (karakterekre) viszi, átjavítja azokat a programnak megfelelően, majd lenyomja a RETURN gombot. Ne felejtse el, hogy az idézőjel időnként megzavarja a VIC-et, ilyen esetben a legcélszerűbb visszamenni a sor elejére és átgépelni az egész sort.

4. Az INST billentyű (SHIFT + INST/DEL együtt) lehetőséget ad arra, hogy karaktereket szúrjon be úgy, hogy ennek a billentyűnek a segítségével a már begépelt szóba vagy sorba üres helyeket tesz.

5. A DELETE billentyű (INST/DEL) simán a kurzortól balra eső karaktert törli.

Térjünk vissza a már futó programokhoz!

Le akarja lassítani a program futását? Nyomja le a billentyűzet bal oldalán található CTRL jelzésű gombot. Ha a CTRL billentyűt lenyomva tartja, a program lelassul. Ime itt egy egyszerű billentyű, amely közli a VIC-kel, hogy milyen gyorsan írjon ki a képernyőre. Igen, de hogy lehet leállítani a program futását? Jó kérdés. Nézzen körül a billentyűzeten mindaddig, amíg meg nem találja a RUN/STOP gombot.
Nyomja le a RUN/STOP billentyűt. A program futása megáll, és a következőt írja ki:

```
BREAK IN 1

READY
```

A VIC nyelvén a *break* azt jelenti, stop. Újra előkerül a kurzor. Észrevette, hogy a kiírás során eltűnt?

Most nézze meg, hogy a programja nem veszett-e el.

Gépelje be a következőt:

LIST és RETURN gomb

Az ön programja (1-es és 2-es sor) megjelenik a képernyőn. Most gépelje be újra a RUN-t, és újra futni fog a program.

Bemutatunk néhány lehetőséget a VIC-en, melyeket a későbbi fejezeteknél gyakran fog használni. Ön már tudja, hogyan kell:

- kiírni a képernyőre egy üzenetet
- törölni a képernyőt (SHIFT/CLR billentyű)
- megírni az első programját (VIC 20) és generálni egy mozgó alakzatot.
- lelassítani a program futását (a CTRL billentyűvel)
- megállítani a programot a RUN/STOP billentyűvel
- programot listázni
- a begépelt sorokat javítani.

Ahogy jobban megismeri a kézikönyv fejezeteit, rengeteg alkalmazását fogja látni az itt elmondottaknak. Ne aggódjon, ha megválaszolatlan kérdések maradtak. Folytassa az ismerkedést, és fokozatosan megválaszolásra kerülnek az eddig még tisztázatlan kérdések. Ezt a kézikönyvet úgy tervezték meg, hogy közvetlenül bármelyik fejezetre ugorhat, mely érdekesnek tűnik. Nem szükséges sorrendben elolvasni minden egyes fejezetet, hogy megismerjük a VIC-et.

2. FEJEZET

A KÉPERNYŐ ÉS A BILLENTYŰZET HASZNÁLATA

Az ön első grafikus karaktere

Kirándulás a VIC-20 billentyűzetén

Nyomtatás a képernyőre

A VIC-20 mint kalkulátor

Bevezetés a színekbe

Gépelje be ezt a programot ugyanígy, és nézze meg, mi történik!

```
10    PRINT "SHIFT CLR/HOME"
20    FOR T = 1 TO 300: NEXT
30    PRINT "Az ön neve"
40    FOR T = 1 TO 300: NEXT
50    GOTO 10
```

Megjegyzés: a 10. sorban a SHIFT és a CLR/HOME billentyűket egyszerre kell lenyomni.
Gépelje be: RUN és nyomja meg a RETURN-t
A program megállításához nyomja meg a RUN/STOP billentyűt.

A KÉPERNYŐ ÉS A BILLENTYŰ HASZNÁLATA

Ez a fejezet feltételezi, hogy elolvasta és megértette az első fejezetben elmondottakat. Amennyiben nem, menjen vissza és olvassa el újra az utolsó két részt belőle, mely bemutatja, hogyan kell használni a billentyűzetet, és hogy kell kiírni a képernyőre.
Indításként helyezze magát kényelembe a VIC billentyűzete előtt, és gépelje be a következőket, beleértve a program sorszámokat és a különböző írásjeleket.
Tartsa lenyomva a SHIFT gombot és nyomja meg a CLR/HOME billentyűt.
Gépelje be: NEW és nyomja meg a RETURN billentyűt.

```
1     PRINT "HELLÓ [SPACE ";        és nyomja meg a RETURN-t
2     GOTO 1                        és nyomja meg a RETURN-t
RUN                                 és nyomja meg a RETURN-t
```

Megjegyzés: Az üres karaktereket a billentyűzet első sorában található hosszú nyomógombbal képezheti. Ez a [SPACE] billentyű.

Miután begépelte a RUN-t, a képernyő megtelik a HELLÓ szóval.

Úgy tűnik, mintha a szavak mozognának oldalirányban. Nyomja meg a CTRL nyomógombot, hogy egy kissé lelassítsa. A VIC kiírja a fenti üzenetet sokszor a képernyő aljáig. Miután megtelt a képernyő, a tartalma elcsúszik felfelé, hogy helyet biztosítson a további nyomtatásnak. Így egy felfelé irányuló mozgás alakul ki. A „barber pole effect" illúzió, melyet a VIC által minden sorban kiírt nagyszámú karakter okoz.

Ha meg akarja állítani a programot, nyomja meg a RUN/STOP gombot.

Most ismét ön következik. Írja be a következő két sort:

```
1     PRINT "HELLÓ NÉV";            és nyomja le a RETURN-t

RUN                                 és újra RETURN
```

Az 1. sorban a NÉV helyett a saját nevét írja be. A pontosvessző azt eredményezi, hogy mindent egymás mellé ír. A mozgás illúziója az üzenet karaktereinek számától függ.

Ha meg akarja állítani a programot, nyomja meg a RUN/STOP billentyűt.

AZ ÖN ELSŐ GRAFIKUS KARAKTERE

Törölje le a képernyőt (tartsa lenyomva a SHIFT-et és nyomja le a CLR/HOME-t.) Most nyomja le a SHIFT és az S gombot. Egy kék szív jelenik meg a képernyőn. Próbálja meg újra. Ön épp most gépelte be az első grafikus karakterét.

Próbáljon más grafikus jelet beírni. Most nyomja le a C= billentyűt, és valamelyik betűt.

Ekkor a billentyű oldalán látható grafikus jelek közül a bal oldali ábra jelenik meg. A bal oldali grafikák jól használhatók pl. táblázatok készítéséhez. Nyomja meg egyszerre a SHIFT és a C= gombot, így nagy és kis betűket is használhat. Magyarázatát lásd a 3. fejezetben.

A KÉPERNYŐ MÉRETE

Mekkora is a VIC képernyője? Találjuk ki! Töröljük le a képernyőt, és gépeljük be a következőt:

```
NEW                                   és nyomja meg a RETURN-t
1     PRINT "♥";                      és RETURN
2     GOTO 1                          és RETURN
RUN                                   és RETURN
```

A kék szív begépelése a SHIFT lenyomása és az S leütésével lehetséges.

A képernyő megtelik kék szívekkel. Számolja meg a szívek számát, amelyek a képernyőn vannak. Minden sorban 22 db-ot talál. Tehát a VIC-nek 22 print pozíciója van a képernyőn keresztben. Ezeket a pozíciókat gyakran oszlopnak nevezik. A VIC-nek 22 oszlopa van.

Hány pozíció van lefelé a képernyőn? Nyomja meg a CTRL billentyűt, hogy lassítsa a nyomtatást. A CTRL billentyű lenyomása az utolsó 4 sor villogását okozza.

A VIC-nek 23 sora van lefelé.

A VIC-nek 506 helye van betűk, szimbólumok, karakterek számára. Tehát egyidőben ennyi karakterrel tud dolgozni a gép.

Állítsa le a szívek kinyomtatását a VIC-en a RUN/STOP gomb lenyomásával. Irja be a VIC-be a következő két sort:

```
1     PRINT "♥";                      és RETURN

RUN                                   és RETURN
```

Ne feledkezzünk el a szív (SHIFT és S) előtti üres karakterről!

A képernyőre kiírt információs formát az idézőjelek közé tett space-ekkel tudja befolyásolni. A másik megoldás, hogy a space (üres karakter) helyett pontot tesz. A fejezet elején található programba a nevén kívül még 3 pontot tegyen és próbálja ki.

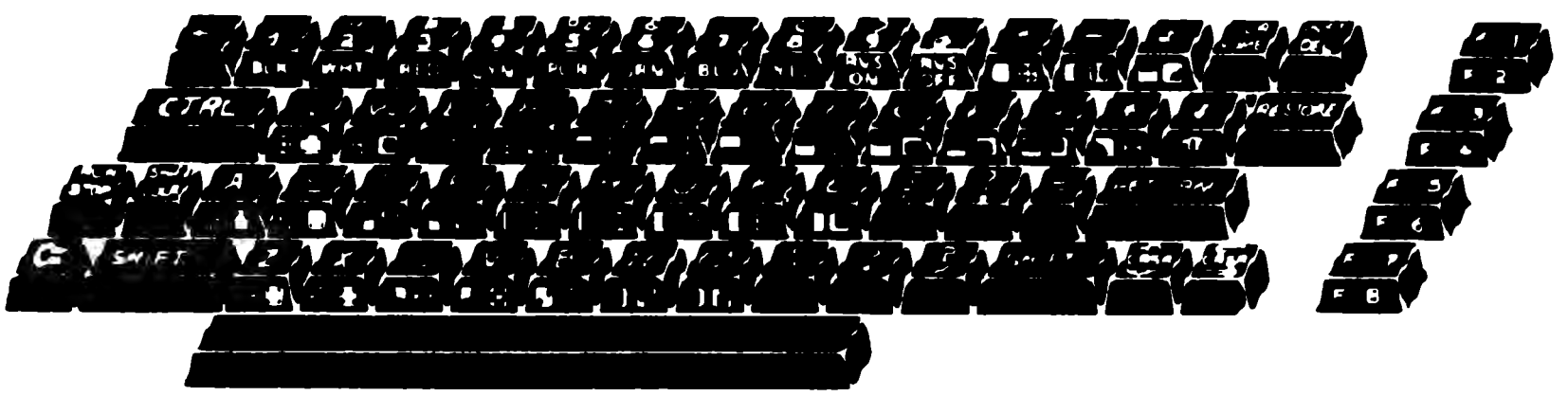
VIC AJÁNLÁS

Amennyiben a VIC 22 oszloppal rendelkezik a képernyőn keresztben, úgy minden olyan üzenet, melynek hossza egész számú osztója a 22-nek (2, 11, 22 karakteres üzenetek), tiszta oszlop formában kiírható. Egyéb hosszúságú üzenetek átcsúsznak a következő sorba. Próbálja ki ön is.

ISMERKEDÉS A BILLENTYŰZETTEL

Ön már használta a billentyűzetet üzenetekhez, grafikus karaktereket íratott ki a képernyőre, vezérelte a VIC működését (CTRL és RUN/STOP), esetleg javította azt, amit már begépelt (INST/DEL).

Itt az ideje, hogy jobban megismerje a billentyűzetet, megtudja, miben is lehet szolgálatára. Nézze meg az alábbi ábrát mely a VIC 20 nagy teljesítményű, sokoldalú billentyűzetét ábrázolja.



Amennyiben a RUN/STOP és a RESTORE billentyűt egyidejűleg nyomja le, alaphelyzetbe állítja a számítógépet (reset), mintha most kapcsolta volna be, azzal az előnnyel, hogy a memóriában levő programot megőrzi, így az listázható, újra futtatható.

Lássuk, hogyan használhatók ezek a billentyűk.

SHIFT **SHIFT billentyűk** — A VIC billentyűzetén, akárcsak az írógépen, két SHIFT és egy SHIFT LOCK billentyű van. A SHIFT billentyűt más billentyűvel együtt használják, hogy grafikus karaktereket nyomtathassanak, vagy pedig kitörölhessék a képernyőt.

CLR HOME

CLR/HOME billentyű – Ennek a billentyűnek a megnyomásakor a kurzor a képernyő bal felső sarkába ugrik. (HOME pozíció). Ha lenyomja a SHIFT billentyűt és ezt a gombot, a kurzor ugyanoda ugrik, de egyben törli a képernyőt.

CRSR

CRSR

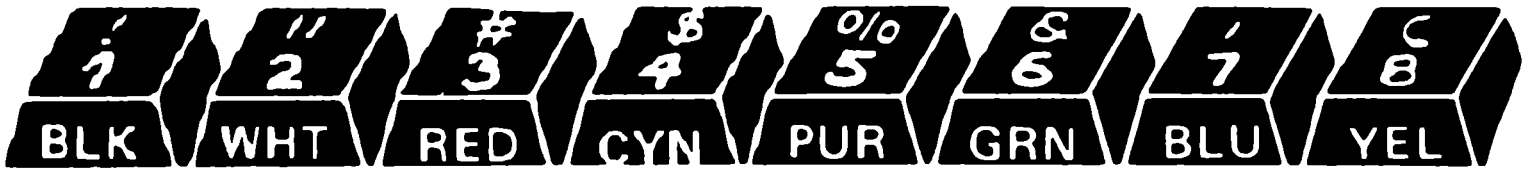
CRSR billentyűk. A VIC segítségével könnyen mozgathatja a kurzort le, fel és oldalirányban. Amennyiben a kurzor-billentyűket lenyomva tartja, a kurzor addig mozog, amíg el nem eresztjük a billentyűt (repeat). Mindegyik billentyűn 2 nyílat találunk, amely megmutatja, hogy milyen irányban – le, fel, oldalt – képes a kurzort mozgatni. Amennyiben a kurzort felfelé vagy balra kívánja mozgatni, úgy a SHIFT billentyűt a megfelelő billentyűvel együtt lenyomva kell tartania. Fontos észben tartani, hogy mozgathatja a kurzort a karaktereken keresztül anélkül, hogy azokra az hatással lenne.

RETURN

RETURN billentyű. Ön megnyomja a RETURN billentyűt minden egyes utasítás végén. Ezzel utasítja a VIC-et, hogy vegyen be egy sort, vagy hajtson végre egy utasítást. Gyakran célszerű a RETURN billentyűt mint ENTER billentyűt felfogni, mert ez beviszi (enter = belép) az utasításokat, információkat a számítógépbe.

CTRL

CTRL billentyű – Ezt a billentyűt a színbillentyűkkel együtt a VIC képernyőjén a színkiválasztáshoz használjuk. A billentyűnek van egy saját ellenőrző funkciója, melyet ön a későbbi felhasználásoknál igénybe vesz. Néhány közvetlen dugaszolható cartidge a Control billentyűt speciális funkciókra is felhasználja. A CTRL billentyű hasonlóan működik a SHIFT billentyűhöz. Lenyomva kell tartani, mialatt lenyomjuk a színbillentyűt.



SZÍN billentyűk – a kijelzendő karakter színét kiválasztja úgy, hogy egyidejűleg lenyomja a CTRL billentyűt és egyikét a 8 szín/szám billentyűnek. A billentyűk előlapján megtalálható minden egyes szín rövidített jelzése. A színek a következők: fekete, fehér, piros, cyan (világoskék), bíbor, zöld, kék és sárga. Ezekkel a billentyűkkel be tudja állítani vagy meg tudja változtatni a program keretén belül, vagy azon kívül a kijelzendő grafika, betű vagy szám színét. Ha egyszer beállít egy színt, mindaddig marad ez a szín, ameddig meg nem változtatja.

RVS ON

RVS OFF

RVS ON és RVS OFF billentyűk. Azt a képet, ahogy a VIC kiír a képernyőre, „megfordíthatja" a CTRL és RVS ON billentyűkkel. Minden, amit ki kíván íratni, fordított (negatív) lesz, ami alatt az értendő, hogy például fehér karaktereket íratthat ki kék háttérrel (amely épp az ellenkezője a normál kiírásnak) a CTRL és a RVS ON billentyűk megnyomásával. A CTRL és RVS OFF segítségével visszatérünk a normál nyomtatási állapotra.

Próbálja ki!

RUN STOP

RUN-STOP billentyű. Nyomja meg ezt a billentyűt és így közli a VIC-kel, hogy álljon meg. A VIC ekkor visszaadja a vezérlést önnek. Amennyiben a VIC-en fut egy program, ezzel a billentyűvel lehet megállítani. Ha lenyomva tartja a SHIFT billentyűt, és beüti a RUN/STOP gombot, a VIC elkezdi az információbetöltést a memóriába a kazettás egységről.

INST DEL

INST-DEL billentyű. Törölni vagy beszúrni lehet a karaktereket ennek a billentyűnek a segítségével. Amennyiben megnyomja ezt a billentyűt, közvetlenül a kurzortól balra levő karakter eltűnik. Ha a sor közepén van, a balra levő karaktert kitörli, és a jobbra levő karakterek eggyel balra csúsznak, hogy kitöltsék az üres helyet. A SHIFT és ennek a billentyűnek a lenyomásával egy space-t, üres karaktert generálunk a sorba, ahová a új karaktert beszúrhatjuk. Ez nagyon hasznos editálásra hibák javítására.

GRAFIKUS KARAKTEREK ÉS COMMODORE BILLENTYŰ — Amikor bekapcsolja a VIC-et ön automatikusan grafikus módban van, mely azt jelenti, hogy nagy betűket és több mint 60 grafikus karaktert gépelhet, melyeket a billentyűkön lát. Ahhoz, hogy a billentyűk jobb oldalán látható grafikát kapja, egyszerűen nyomja le a SHIFT gombot, valamint azt a billentyűt, amelyen a kívánt grafika látható. Ahhoz, hogy a bal oldali grafikát kapja meg, tartsa lenyomva a COMMODORE billentyűt (a kis zászló). Ily módon a nagy betűket és a teljes grafikus készletet kigépelheti egyszerre, ábrákat, táblázatokat készíthet oly módon, hogy a karaktereket egymás mellé, fölé helyezi (mintha blokkokat építene).

UPPER/LOWER CASE and GRAPHIC billentyűk.

Amennyiben a SHIFT és COMMODORE billentyűket egyszerre nyomja le, a VIC-et text (szöveg) módba állítja. Ez esetben úgy használhatja a VIC-et, mint egy normál írógépet a teljes kis- és nagybetűs készlettel, valamint a billentyű bal oldalán látható karakterekkel. A baloldali grafikus lehetőségek ideálisak táblázatok, grafikák, formanyomtatványok készítésére. Ahhoz, hogy visszakerüljön a „Nagy betű/teljes grafika" módba nyomja meg a SHIFT és COMMODORE billentyűket egyszerre.

PROGRAMOZHATÓ FÜGGVÉNY-BILLENTYŰK

Négy billentyű a konzol jobb oldalán nem definiált, amikor ön bekapcsolja a VIC-et. Ezeket különböző függvényekhez, feladatokhoz rendelheti a felhasználás során, melyeket ön határozhat meg. A SHIFT billentyűvel vagy anélkül 8 féleképp használhatjuk a függvénybillentyűket. Elsősorban dugaszolható Cartridge-ok speciális programjaihoz használhatjuk, de a számítógép programozó ugyanúgy használhatja ezeket a billentyűket.

SPECIÁLIS BILLENTYŰK.

A VIC billentyűzete tartalmaz olyan speciális szimbólumokat, melyek nem találhatók meg minden írógépen, vagy még inkább a számítógépek többségén. Például: az angol font (£) jele, a pí (π) nyíl vissza (←) nyíl fel (↑), kisebb nagyobb (><) és a zárójel ([ ])

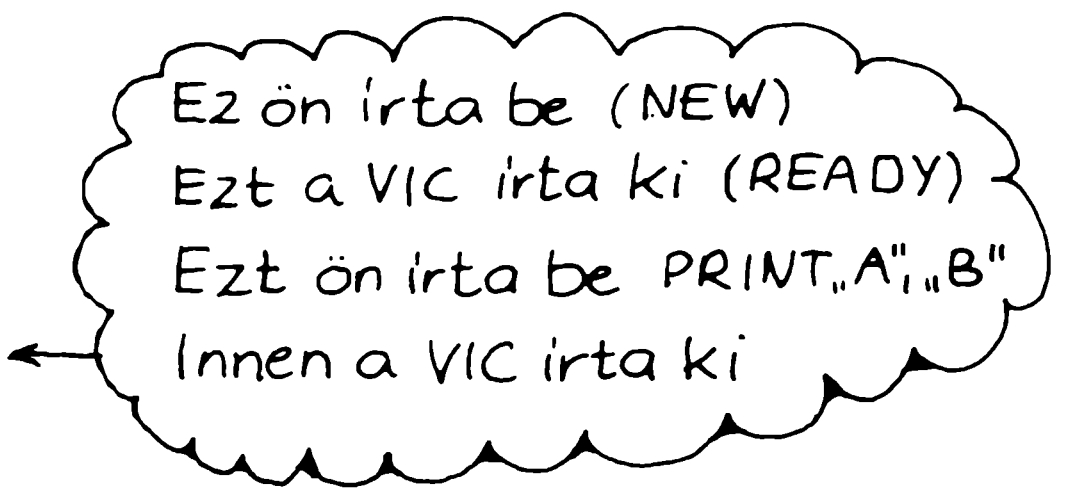
Ezzel befejezzük a VIC billentyűzet ismertetését. A legjobb módszer a megismerésére, ha elkezdünk „próbálkozni", kísérletezni rajta. Gyakorlás a billentyűzeten. Próbálja ki a kis/nagy betűs üzemmód fent említett lehetőségeit. Meglátja, milyen ügyes dolgokat tud csinálni a VIC grafikus készletével. A billentyűzet az ön közvetlen kapcsolata lesz a VIC-hez. A billentyűzet ismerete segít a VIC megismerésében.
Törölje a képernyőt, és adja be a következő sorokat:

NEW és RETURN gomb

PRINT "A", "B" és RETURN gomb

A képernyő a következőképp néz ki:

```
NEW
READY
PRINT "A", "B"
A               B
READY
```



Most írja be ezt a sort, és nyomja meg a RETURNT:

PRINT "A"; "B"

Ezután a képernyő így néz ki:

```
NEW
READY
PRINT "A", "B"
A               B
READY
PRINT "A"; "B"
AB
READY
```

Amennyiben vesszőt használ az első PRINT utasításban, a VIC kiírja a betűket a képernyőre, de elválasztja őket több üres hellyel. Amennyiben pontosvesszőt használ, a VIC a 2 betűt egymás mellé írja.

Az első esetben a két betű között pontosan 11 karakter a távolság. Ez a tény megmutatja, mi is történik tulajdonképp. A VIC a képernyőt két egyenlő részre bontja. Amikor a VIC vesszővel elválasztott szöveget vagy számokat ír ki, ekkor az első kiírandó szöveget a képernyő bal oldalára, a másikat a jobb oldalára írja, hacsak az első szöveg nem hosszabb, mint 11 karakter.

Amennyiben az első szöveg kisebb vagy egyenlő, mint 11 karakter, a VIC kiírja ezt, elmegy a képernyő közepére, és onnan kezdve kiírja a másodikat.

Amennyiben az első szöveg hosszabb, mint 11 karakter, a második a következő sorban kezdődik. Törölje a képernyőt, és próbálja ki ezt a példát.

```
PRINT "ABCDEFGHIJKL", "X"
```

A képernyő a következőképp fog kinézni:

```
PRINT "ABCDEFGHIJKL", "X"
ABCDEFGHIJKL
X
READY
```

Ebben az esetben az üzenet első fele 12 karakter hosszúságú, így az "X" a következő sorba kerül. Ismételje meg a feladatot pontosvesszővel (;) a két szöveg között.

```
PRINT"ABCDEFGHIJKL"; "X"
```

Ugyanezt az eredményt mutatja az ön VIC-je?

```
PRINT ABCDEFGHIJKL"; "X"
ABCDEFGHIJKLX
READY
```

Megjegyzés: Az X most a többivel egy sorba került.

Érti a logikáját? A VIC úgy működik, mint egy írógép automatikus tabulátorral a képernyő közepén. Ha vesszőt lát, vagy a képernyő közepére vagy az elejére tabulál, attól függően, hogy melyik érhető el hamarább.

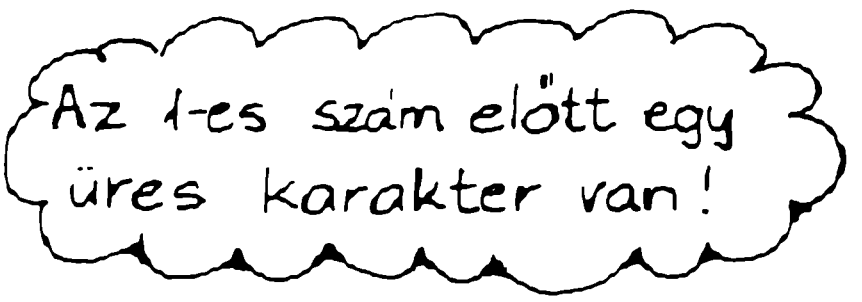
Törölje a képernyőt és gépelje be a következő sort a VIC-be.

```
PRINT 1, 2
```

Számok esetén az idézőjel elhagyható.

A képernyő a következőképp néz ki:

```
PRINT 1, 2
  1        2
READY
```



Látja az üres karaktert az első szám előtt? Amikor a VIC kiír egy számot, hagy egy üres helyet előtte az előjel számára. Amennyiben a szám negatív, a mínusz előjel jelenik meg a képernyőn.

Próbálja ki, nézze meg. Gépelje be a következő sort a VIC-be:

```
PRINT −1, −2
```

Nézze meg a képernyőt, mi került kiírásra.

```
PRINT -1, -2
-1          -2
READY
```

Megjegyzés: A számok azonos helyen vannak.

A számok ugyanazon a helyen jelennek meg, mint az előző példában. Előttük a negatív előjel található.

Ez a néhány példa ötleteket ad arra, hogyan kaphatja vissza a VIC-től az átadott üzeneteket, információkat a képernyőre.

A VIC KALKULÁTOR

A VIC-et úgy is használhatjuk, mint 9 digites kalkulátort. A + és – előjelek úgy használhatók, mint a matematikában. A VIC szorzójel a csillag (*), az osztásé pedig a ferde vonal (/).

Gépelje be ezeket a számításokat, és ellenőrizze az eredményt. A C) és K) Mellékletben még további információk találhatók.

Megjegyzés: A matematikai ferde vonal (osztás jele) a ? billentyűnél van. Az N billentyűn lévő ferde vonal egy grafikus szimbólum, és nem az osztás jele.

```
PRINT 1 + 1                    RETURN gomb
PRINT 3–2                      RETURN gomb
PRINT 5 * 2                    RETURN gomb
PRINT 6/3                      RETURN gomb
PRINT 2 * (4/2)                RETURN gomb
PRINT 5000/5                   RETURN gomb
PRINT 2/3                      RETURN gomb
PRINT 3 ↑ 3                    RETURN gomb
```

Megjegyzés: a ↑ jel a hatványozás jele. Azt jelenti: 3^3, vagy 3 x 3 x 3

Ha ki akarja íratni a számítás eredményét (PRINT), annak az idézőjelen kívül kell lennie.

Próbálja ki ezeket a példákat:

```
1 PRINT" 2 * (4/2)"                    RETURN
```

Ekkor a VIC kiír mindent az idézőjelen belül

```
1 PRINT "A VÁLASZ" 2 * (4/2)           RETURN
```

BEVEZETÉS A SZÍNEKBE

A VIC betűket, számokat, grafikus szimbólumokat ír ki 8 különböző színben. Ezenkívül képes negatív karaktereket is kiírni.

Törölje a képernyőt, majd nyomja le a CTRL gombot, valamint az RVS ON-t. (Ez a gomb, amelyiknek a tetején a 9-es szám van.) Most eressze el a CTRL gombot, és tegye a kezét a SPACE-re (hosszú billentyű alul). Tartsa lenyomva. Mi történik? Egy kék csíkot rajzol keresztbe a képernyőn.

Tartsa a SPACE-t annyi ideig lenyomva, ameddig csak akarja. A kurzor eltűnik a képernyő jobb szélén, és majd előkerül a bal oldalon, és egy nagyobb kék csík keletkezik.

Engedje el a SPACE billentyűt, és tegye a következőt. Tartsa lenyomva a CTRL billentyűt, és nyomja meg a RED (a tetején a 3-as szám van) gombot.

A kurzor ekkor piros lesz. Tartsa ezek után a SPACE gombot lenyomva. Most piros csík lesz? Igen. Folytassa a festést.

Cserélje a színeket kívánsága szerint. Olyan vastaggá teheti a színes csíkokat, amilyenné kívánja.

Most üsse be a CTRL RVS OFF-t és nyomja le a SPACE-t. Semmi sem történik, mindössze üres karakterek lesznek. Nyomja be a CTRL RVS ON-t, és a szín visszatér. Próbáljon ki néhány betűt negatívban. Az így nyomtatott betűk kiválóak fejléchez, vagy kiemeléshez. Negatív karaktereket használhat a programon belül is. Például próbálja ki ezt:

```
NEW
10 PRINT " CTRL  RVS/ON  VIC 20"
20 GO TO 10
RUN
```

Megjegyzés: a 10. sorban a CTRL és az RVS/ON gombot egyszerre kell lenyomni. Ne feledkezzen meg minden sor végén a RETURN-ről!

Hogy a gép készen álljon a következő fejezet kísérleteihez, törölje ki a szükségtelenné vált programot. Nyomja le egyszerre a RUN/STOP és a RESTORE gombot, majd gépelje be a NEW szót, végül nyomja meg a RETURN billentyűt.

3. FEJEZET

SZÍN ÉS GRAFIKA

Programozás színesben

A VIC színes billentyűk

Alap és háttér színcseréje

Alap és keret színkombinációk

Képernyő színezése

Képernyő memóriahelyek

Véletlen színek

Hang- és színkombinációk

Billentyűzet grafikák

Fejlécek, címek grafikája

Gépelje be ezt a programot pontosan úgy, ahogy itt látja, és nézze meg, mi történik.

```
1     FOR H = 1 TO 505
2     PRINT"♥"
3     NEXT
4     FOR C = 8 TO 255 STEP 17
5     POKE 36879,C
6     FOR T = 1 TO 500: NEXT
7     NEXT
8     GOTO 4
RUN
```

Megjegyzések: a program első 2 sora hatására 505 szívet ír ki a képernyőre. A 4 sorban C értékét 17-es lépésközzel változtatja. A 6. sor időkésleltetéses ciklus, mely hatására a VIC elszámol 500-ig, mielőtt újra megváltoztatná a színt.

A program megállításához nyomja meg újra a RUN/STOP billentyűt.

SZÍN ÉS GRAFIKA

A VIC és az ön színes televíziója biztosítja azt a lehetőséget, hogy a képernyőn bárhova színeket tegyen. Amikor ön először bekapcsolja a VIC-et a keret, a kurzor és a karakterek a képernyőn már színesek. De ez csak a kezdet. A VIC 8-féle kurzor, 8 keret és 16 alap (képernyő) színt képes produkálni.

Kezdjük azzal, hogy a VIC-en található billentyűket arra használjuk, hogy színeket produkáljunk a képernyőn. Írjunk be valamilyen betűt. A betű a fehér képernyőn kék színben jelenik meg. Most tekintsünk a felső soron levő billentyűkre (1-től 8-ig vannak számozva). Látja a színek nevét, melyeket a billentyűk elejére írtak. Most keresse meg a billentyűzet bal oldalán található CTRL billentyűt. Tartsa lenyomva a CTRL billentyűt, és nyomja le azt a gombot, melyen a YEL (sárga) jelzés van. Engedje el mind a két gombot, és üssön le egy betűt a billentyűzeten. Ez a betű sárga színű lesz. Most tartsa lenyomva a CTRL billentyűt és üssön le egy másik színt, majd gépeljen be néhány betűt. Látja, milyen egyszerű a betűk színét cserélni a képernyőn? Most keresse meg az RVS/ON jelölésű gombot. Nyomja le a CTRL billentyűt és az RVS/ON-t is. Próbáljon meg begépelni néhány betűt. Minden betű, amit begépel (kivéve a RETURN), "reverse" módon jelenik meg a képernyőn, mint egy fotó negatívja. Amennyiben lenyomja a CTRL és RVS/OFF billentyűket, a betűket normál állapotban jelzi ki.

PROGRAMOZÁS SZÍNESBEN

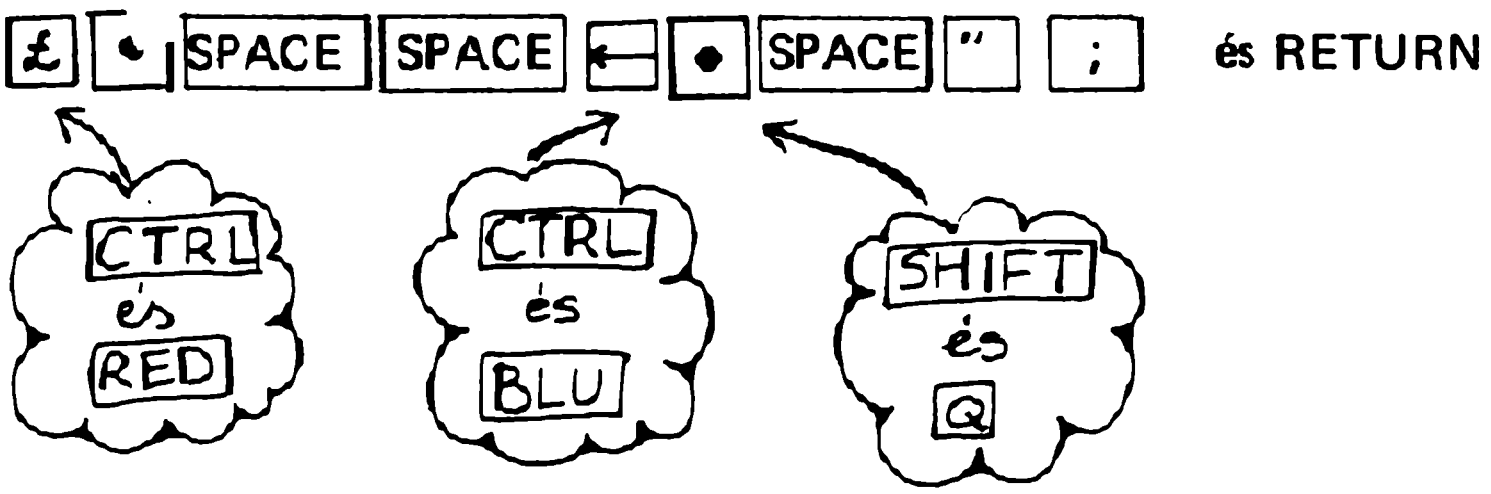
Most kombináljuk a színek vezérlését egy egyszerű program-utasítással. Ne feledje, ha ön a CTRL-t és a színbillentyűt idézőjelen belül gépeli, fordított grafikát kap. Most tegye a követ-

kezőt. Tartsa lenyomva a SHIFT billentyűt, és üsse le a CLR/HOME-ot. Gépelje be a következőket:

NEW és nyomja meg a RETURN gombot

Ezután gépelje be a következőket:

1 PRINT" [SPACE]



és RETURN

Ezután folytassa így:

2 GOTO 1 és RETURN gomb

Ha gondjai vannak ennek a példának a begépelésével, lapozzon vissza a „A kezdet" című részhez az első fejezetben.

Nem lehet kárt tenni a VIC-ben, bármit gépel is be, de bizonyos billentyűkombinációkkal megzavarhatja a gépet. Ha véletlenül lenyomva felejtette a SHIFT/LOCK billentyűt, a displayre kiírtakat nehéz lesz kibetűzni. Ha hibát követ el, nyomja le egypárszor a RETURN billentyűt, és gépelje be újra az egész sort. Az új sor automatikusan eltünteti a régit. Miután rendben megvan a két sorral a képernyőn, üsse be:

RUN és nyomja meg a RETURN-t

Ha végül megnyomta a RETURN-t, a képernyő tele lesz piros és kék labdák százaival. Hogy lehetséges ez? Ez nagyon egyszerű a VIC-kel. Nézze meg a fenti példát. Látja, milyen két furcsa karakterrel kezdődik a sor? Ezek akkor keletkeztek, amikor lenyomta a CTRL billentyűt és a RED vagy BLUE (piros vagy kék) billentyűket. Ezek a szimbólumok a VIC rövidítésével azt jelentik, hogy az első labda piros, a második kék legyen.

Gépelje be a következőt:

[£] – ezt úgy kapja, hogy miközben lenyomva tartja a CTRL-t, nyomja le a RED billentyűt.

[←] ezt pedig úgy kapja, hogy a CTRL mellé a BLUE billentyűt nyomja le.

Semmi az egész! Amennyiben belefáradt a pirosba és a kékbe, nyomja meg a RUN/STOP gombot. Ha szeret önálló felfedezéseket tenni, írja át az 1. sort. Használja a CTRL-gombot és a színbillentyűket grafikus karakterekkel, betűkkel, számokkal.

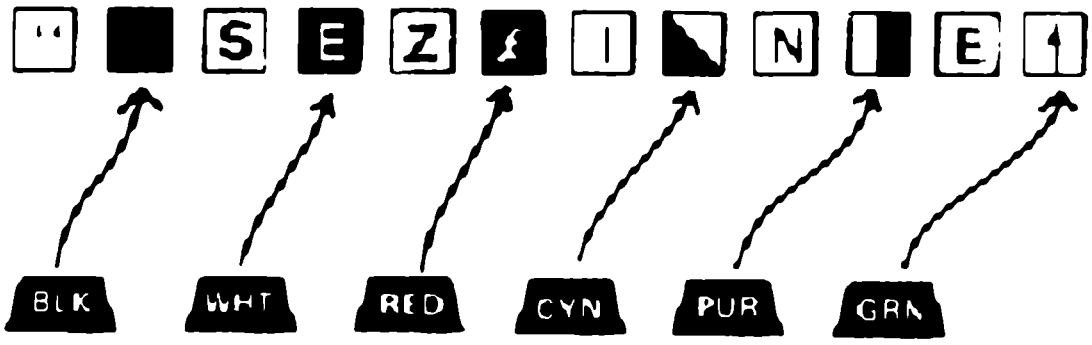
A VIC SZÍNBILLENTYŰK

Az előző példából kiderült, hogy egy PRINT üzenetbe szín vezérlést lehet betenni a CTRL és a következő címkéjű billentyűk segítségével:

BLK (fekete), WHT (fehér), RED (piros), CYAN (halványkék), PUR (bíbor), GRN (zöld), BLU (kék), YEL (sárga)

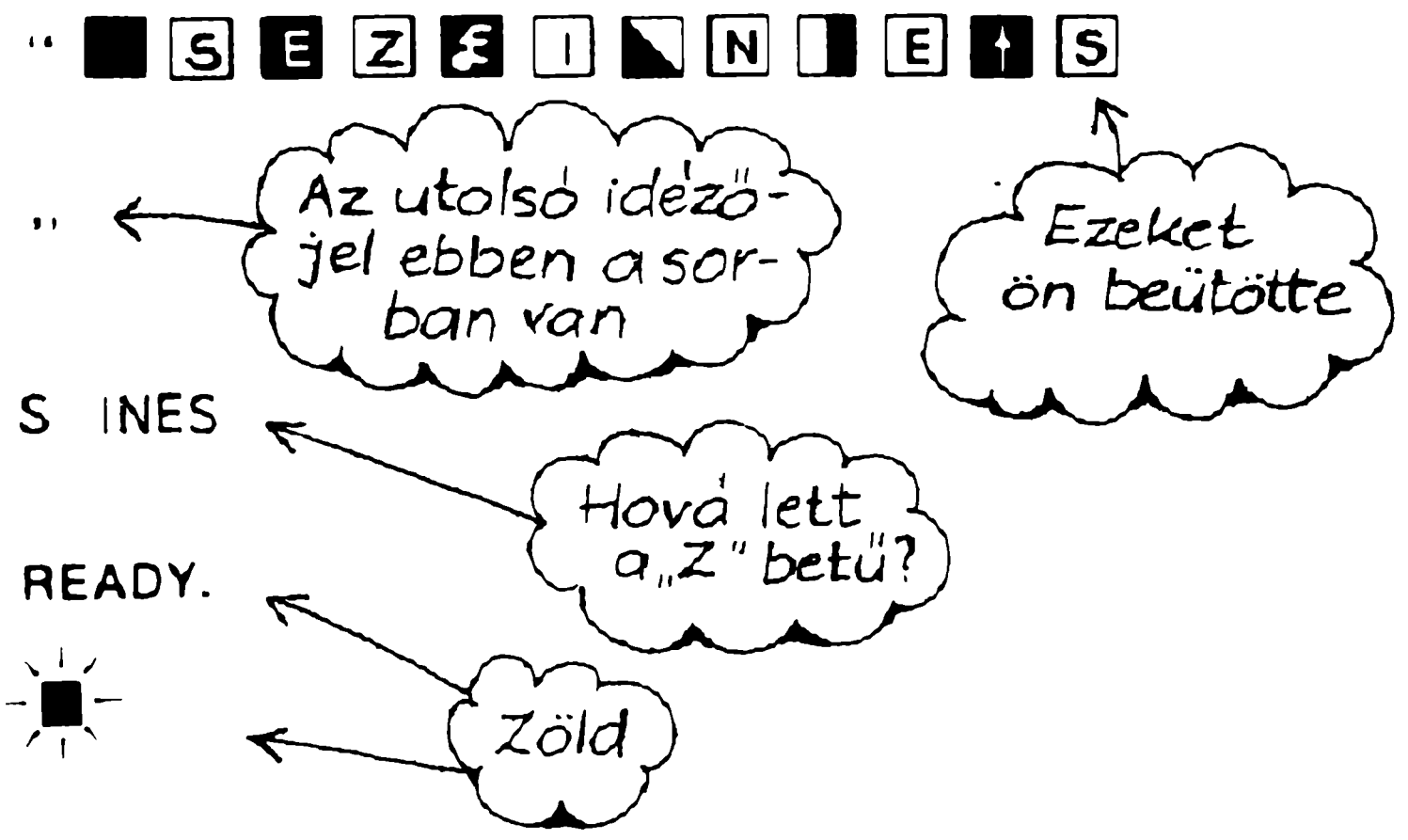
Ezek a billentyűk ugyanazok, mint a számbillentyűk. Ha ezeket a gombokat a PRINT üzenet folyamán nyomja le, furcsa karakterkészlet jelenik meg. Hogy lássa, hogy néznek ki ezek a karakterek, nyomja meg a RUN/STOP billentyűt (amennyiben a program még fut), törölje a képernyőt, és adja be:

```
PRINT
```



Ne feledkezzen meg az idézőjel és a CTRL billentyűk használatáról a színbillentyűknél. A képernyő így fog kinézni:

```
PRINT
" SEZ INES
READY.
```



Hol van a Z betű? A második színbillentyű fehér, és a háttér is — ahogy sejti — fehér.

A fehér Z betű a fehér háttérben a SZÍNES szóban egy üres helyet eredményez.

A további négy betű a SZINES szóban mind különböző színű. Az utolsó betű az "S", zöld. Figyelje meg, hogy a READY üzenet szintén zöld a kurzorral együtt.

Amennyiben színvezérlés van egy PRINT üzenetben, a VIC emlékszik az utolsó használt színre, és megtartja azt a színt. Ha a kurzort vissza akarja változtatni kék színűre, nyomja le a CTRL-t és a BLU billentyűt együtt. Ha kedve van, próbáljon meg néhány saját tervezésű színes üzenetet. Rövid időn belül a színvezérlő billentyűk további használatába tekinthet bele.

HÁTTÉR- ÉS KERETSZÍNEK CSERÉJE

Most, hogy már tudja, hogyan lehet cserélni a betűk, ábrák színét, itt az ideje, hogy megismerje a képernyő (háttér) keret színeinek a cseréjét. 255 képernyő — keret színkombináció van. Azért, hogy biztos legyen abban, hogy a VIC készen áll, nyomja meg a RUN/STOP, majd a SHIFT és a CLR/HOME billentyűket a képernyő kitörlése végett. Ezután gépelje be a következő sorokat a VIC-be:

```
NEW                    és RETURN
1    FORX = 1 TO 255
2    POKE 36879, X
3    PRINT" ♥  POKE 36879,"X
4    FOR T = 1 TO 100: NEXT T
5    NEXT X
```

Megjegyzések: A 4. sor időkésleltetéses ciklus. A VIC 1-től 1000-ig számol a változások között. 5. sor hozzáad 1-et X-hez — más szavakkal megváltoztatja X-et a következő számra, és mindent újra elvégez.

Miután begépelte ezeket a sorokat, nézze át, hogy megegyeznek-e az itt bemutatottal. Ha nem, akkor gépelje be a hibás sorokat újra az elejétől. Használhatja a CRSR-t, hogy vissza tudjon térni arra a sorra, melyet át akar írni. Miután minden sorral meg van elégedve, vegyen egy darab papírt, tollat vagy ceruzát és helyezze a keze ügyébe. Majd gépelje be:

RUN és nyomja meg a RETURN-t

Ezután a képernyő elkezd pirulni és villogni. A háttér megváltoztatja a színeit. Még a kis üzenetek a képernyő tetején is megváltoztatják a színűket. A VIC kiír 255 különböző színkombinációt.
(Syntax error — szintaktikai hibát jelez az egyik sorban? Írja újra a sort, majd üsse be: RUN és RETURN újra).

Miközben a VIC fut, és ön észrevesz egy érdekes színt, színkombinációt, írja fel a számát a papírra. Csak a kiírt üzenet végén levő szám változik. A POKE 36879 változatlan marad.

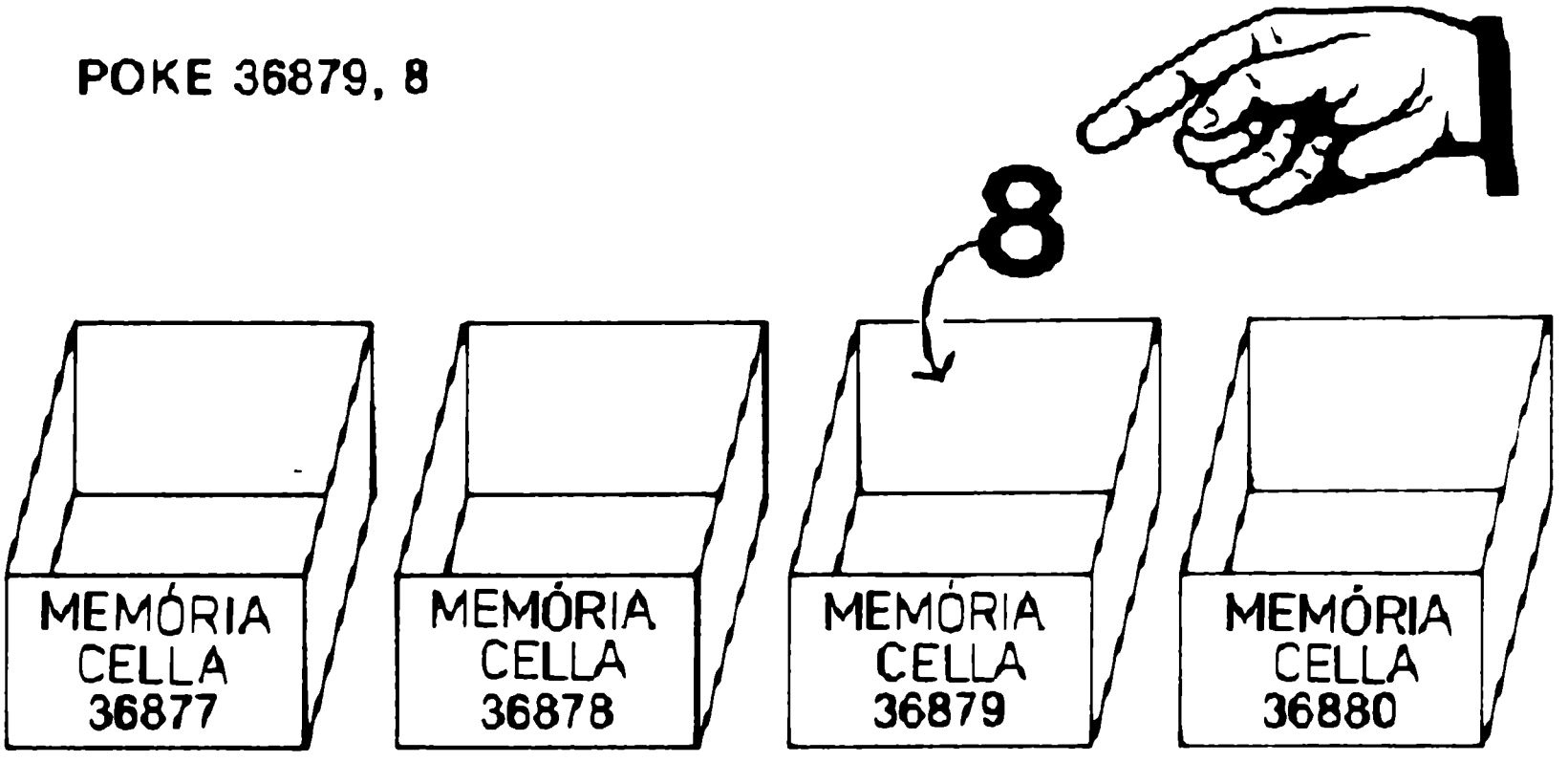
(Esetenként nem tudja elolvasni a szöveget, mert azonos a színe a hátterével. Ilyen az élet. Néhány felvillanás után a számok és a betűk visszatérnek. A VIC-nek 8 keretszíne, 16 alap- (tulajdonképpen ez a háttér) és 8 karakterszíne van. A 8 különböző színű karakter-lehetőség közül bármelyiket bármely színű alapra írhatja. Ez rengeteg próbálkozási lehetőséget rejt.)

Amennyiben a képernyőt eredeti színére akarja visszaállítani, egyszerűen nyomja le a RUN/STOP billentyűt, és mellé a RESTORE-t.

A fenti példa 2. sora felelős a VIC színek állításáért. Ez a sor tartalmazza a POKE utasítást. Minden POKE utasításnak két numerikus értéke van, melyet a VIC használ: POKE 36879, X

Az első szám (a mi esetünkben a 36879) azt a memóriacímet takarja, ahová akarjuk tenni (gondoljon egy kis dobozra, melyet 36879-es címkével láttak el), a második a számot (X). A 36879 az a memóriacím, ahol a képernyő keret- és alapszíneinek a meghatározása történik. Minden X érték egy különböző színkombinációnak felel meg. Ebben a példában X az 1 értékkel indul, és változtatja ezt az értéket 2, 3-ra egészen 255-ig.

Az alábbi ábra a POKE 36879,8-t ábrázolja.



Hogy segítsünk önnek a megfelelő színkombinációk kiválasztásában, bemutatunk egy táblázatot a POKE értékek és a színkombinációk kapcsolatáról. A táblázat tartalmazza a színkombinációk előállításához szükséges POKE értéket. Ezek soronként 8-as ugrással emelkednek. A hiányzó számok azok a POKE értékek, melyek a „negatív" karaktereket eredményezik.

ALAP	HÁTTÉR							
	Fekete	Fehér	Piros	Cián	Lila	Zölt	Kék	Sárga
Fekete	8	9	10	11	12	13	14	15
Fehér	24	25	26	27	28	29	30	31
Piros	40	41	42	43	44	45	46	47
Ciánkék	56	57	58	59	60	61	62	63
Lila	72	73	74	75	76	77	78	79
Zöld	88	89	90	91	92	93	94	95
Kék	104	105	106	107	108	109	110	111
Sárga	120	121	122	123	124	125	126	127
Narancs	136	137	138	139	140	141	142	143
Vil. narancs	152	153	154	155	156	157	158	159
Rózsaszín	168	169	170	171	172	173	174	175
Vil. ciánkék	184	185	186	187	188	189	190	191
Vil. lila	200	201	202	203	204	205	206	207
Vil. zöld	216	217	218	219	220	221	222	223
Vil. kék	232	233	234	235	236	237	238	239
Világossárga	248	249	250	251	252	253	254	255

A KÉPERNYŐ SZÍNEZÉSE

Itt az ideje, hogy összekombináljuk a VIC szín- és grafikus tulajdonságát egy kiváló színes kiíratásra. Újra be kell adnia néhány sort a VIC-be, és érdekes dolgok fognak történni. Ha valamilyen példa még ott kódorog a képernyőn, a RUN/STOP-pal állítsuk meg. Töröljük a képernyőt, és írjuk be a következő misztikus sort:

```
NEW
1       L = INT (RND (1) * 500) + 1
2       C = INT (RND (1) * 8) + 1
3       POKE 7680 + L,160
4       POKE 38400 + L,C
5       GOTO 1
```

Megjegyzések: Az 1. sor egy véletlen számot generál 1 és 500 között. A 2. sor egy véletlen számot generál 1 és 8 között. Az INT, RND utasításokról a 7. fejezetben olvashat.

Ellenőrizze, hogy minden sort helyesen gépelt-e be. Ha minden rendben, törölje a képernyőt, és gépelje be: RUN. A képernyő különböző színekbe borul.

Ezt szintén a VIC csinálta. MIndössze néhány billentyű lenyomásával mindenfelé színek vannak. Hogyan lehetséges ez?

Röviden: az 1. és 2. sor véletlenszerűen kiválasztja, milyen szín (C = color — szín) a képernyőn (L = location — hely), hová kerüljön. Az 1. sor számokat generál 1 és 500 között, a 2. sor pedig 1 és 8 között. A 3. sor kiteszi a karaktert a képernyőre. A 7680-as értékek az ún. "home" pozíciót jelöli a képernyőn — a bal felső sarkot. A POKE 160 érték egy fix blokk (éspedig a "negatív space").

A 4. sor adja meg a karakterek színét. A 38400-as érték a home pozícióban megadja a karakter színkomponenseinek a memória címét.

KÉPERNYŐ CÍMEK

VIC AJÁNLÁS:
Ahhoz, hogy karaktereket kivigyen a képernyőre, meg kell adni a képernyő címét és a szín címét, minden egyes karakternek. A képernyő-címek 7680-nál kezdődnek, a színek pedig 38400-nál. Lásd a képernyő memória térkép függeléket.

Lehet más értékeket is a képernyőre kivinni. Cserélje ki a 3. sorban levő 160-as értéket bármilyen másra 0 és 255 között.

506 lehetséges képernyő-cím létezik. Ön bármilyen szót, betűt, mondatot, grafikát tud íratni a képernyőre. Csak annyit jegyezzen meg, hogy a képernyő 506 részből áll, és minden részhez tartozik egy szám.

Elég a POKE-olásból, térjünk vissza a színvezérlő billentyűkhöz.

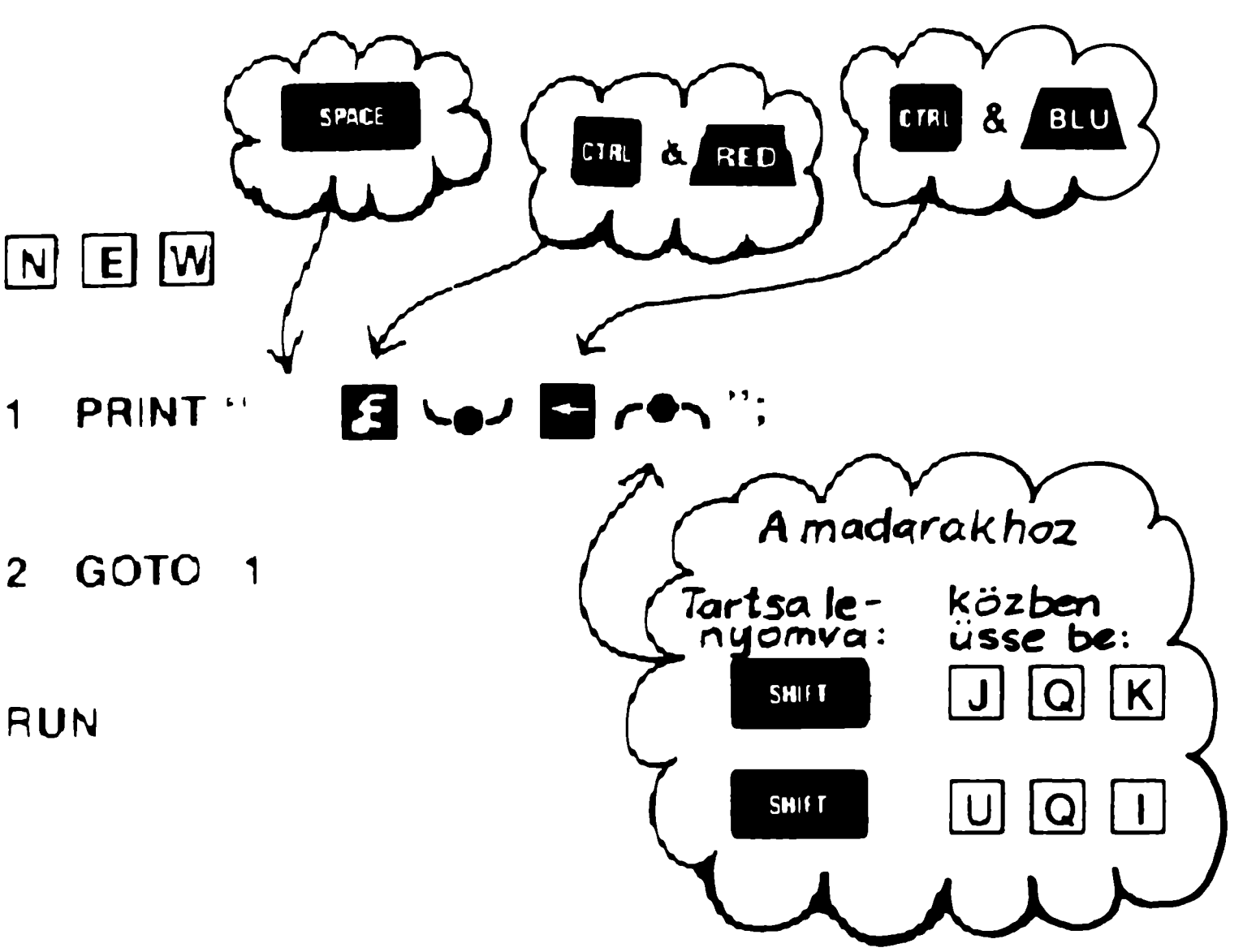
De előbb ahhoz, hogy visszaállítsa a VIC-et alapállapotba, üsse be a következő POKE utasítást:

POKE 36879, 27 és RETURN

VÉLETLEN SZÍNEK

Most megismeri, hogyan lehet a VIC-re hagyni a színek kiválasztását.

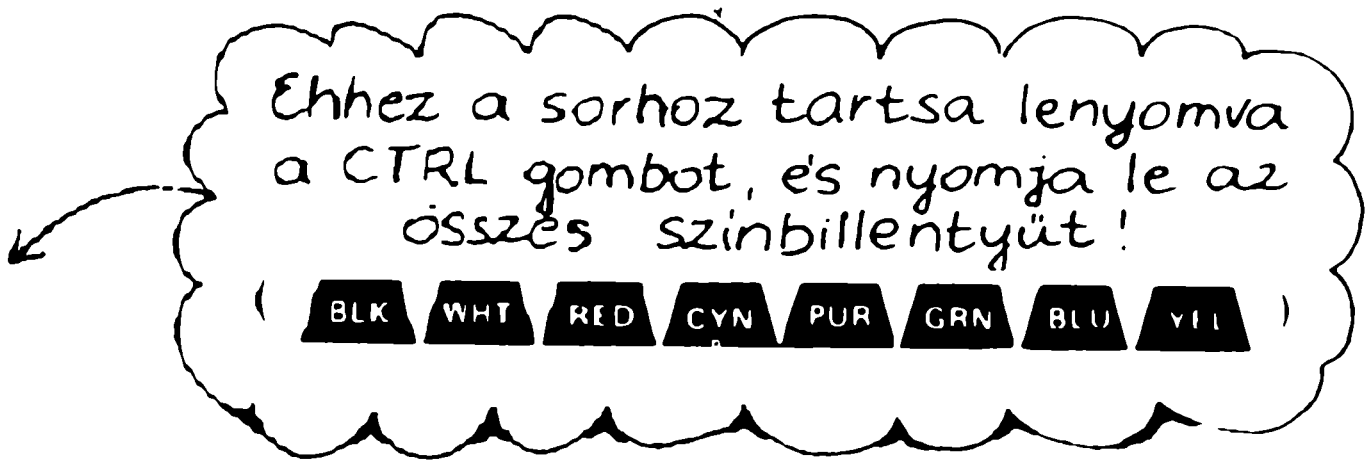
Elsőként törölje a képernyőt és üsse be a következő sorokat (a sorvégi RETURN-ről ne feledkezzen meg)



Így a képernyőt feltölti piros és kék madarakkal.

Hagyja a madarakat repülni egy ideig. Ha meg akarja állítani őket, nyomja meg a RUN/STOP gombot.

Most, hogy már tudja, hogyan kell madarakat készíteni, törölje a képernyőt, és üsse be a következő sorokat:
(Ne feledkezzen el a RETURN-ről)



N E W

```
1  A$ = "
2  N = INT (RND (1) * 8) + 1
3  B$ = MID$ (A$, N, 1)
4  PRINT B$ "
5  GOTO 2
```

Megjegyzések: A 2. sor egy véletlen számot generál 1 és 8 között. A 3. sor — színvezérlés — véletlen érték — kiválaszt egy színt véletlenszerűen az A$ színvezérlőkből. A 4. sor elején ügyeljünk az idézőjel utáni egy üres karakterre. (SPACE-billentyű)

Ellenőrizze még egyszer a begépelt sort, hogy megegyezik-e a fentiekkel. Ha készen van, törölje a képernyőt, és üsse be a RUN-t. Amennyiben hibajelzés történik, írja újra a hibás sort, majd üsse be a RUN-t újra. Madarak, madarak mindenütt. A szivárvány minden színében. A képernyőn ott döngicsél egy falka szivárványszínű madár. Figyelje meg, hogy a raj hiányos. A VIC oda valójában fehér színű madarat rajzolt. Ha ezeket a láthatatlan madarakat láthatóvá kívánja tenni, meg kell változtatnia a hátteret. (Emlékszik rá, hogyan? Egy számot kell a 36879-es címre POKE-olni, beírni.)

Természetesen ha a hátteret egy másik színre cseréli, mondjuk kékre, akkor a kék madár tűnik el. Próbálja ki, győződjön meg róla. Nyomja meg a RUN/STOP-t ha úgy gondolja, hogy a madarak hagyják abba a repülést.

Az utolsó példa a VIC néhány érdekes tulajdonságát használja fel a madarak készítésénél és azok színváltoztatásánál. Nézzük meg a példát soronként. Az első sor:

```
1  A$ = "■ E ε ◥ ◨ ↑ ← π "
```

A színvezérlőket egy hosszú karakterláncként (string) beírjuk a VIC A$-nek nevezett memóriába (szóban *„A stringnek", „A dollár"-nak* nevezzük). Az A$-t úgy képzelje el, mint egy dobozt, melyben karakterláncokat lehet tárolni. Az A$ jelű doboz első pozíciója a BLK (fekete) vezérlőkaraktert tartalmazza. Az utolsó pozíció a YEL (sárga) vezérlőkaraktert tartalmazza.

A következő sor:

```
2      N = INT (RND (1) * 8) + 1
```

Ez a sor véletlenszerűen állít elő egy pozitív egész számot 1 és 8 között, és beteszi N-be. Az N egy olyan hely a VIC memóriájában, amelyet számok tárolására lehet használni. A VIC onnan tudja, hogy N-et számok tárolására használhatja, hogy nincs dollárjel a név végén. Az első sorban azt a helyet, ahol a színvezérlőket tárolják, A$-nek hívták, és egy dollárjel volt a végén. A VIC tudja, hogy azokat a helyeket, melyek nevei dollárjellel végződnek, üzenetek, karakterláncok tárolására használják. Arról, hogy az INT és RND működésével kapcsolatosan hogyan állítják elő a véletlen számokat, további információkat a könyv 7. fejezetében talál. Számunkra most elég, hogy ez a sor 1 és 8 között számokat generál. Nézze meg a 8-as számot az egyenlőség jobb oldalán. Ez a szám határozza meg, melyik legyen a legnagyobb generált véletlen szám. Ha kicseréli 8-ról 6-ra, akkor 1 és 6 közötti számokat tud képezni. Most pedig menjünk tovább:

```
3      B$ = MID$ (A$, N, 1)
```

Ez egy új, B$-nak nevezett „dobozt" képez, és 1 színvezérlő karaktert tesz bele. Az A$ az összes színellenőrző karaktert tartalmazza. N egy véletlenszám 1 és 8 között. És mit tesz a MID$? Ez kiveszi az "N"-edik pozíciójú színvezérlő karaktert. Ezt a karaktert beteszi B$-be. Most már a VIC-en belül 3 kis dobozunk van, melyeket A$-nek, "N"-nek és B$-nek hívnak. Minden alkalommal, amikor a VIC eljut a 2. sorba, új értéket kap N. Ez meghatározza, hogy a MID$ milyen színt vesz ki és tölti a B$-ba.
A negyedik sor:

4 PRINT B$ " └┘ ┌┐ "

Ez a sor megmondja a VIC-nek, hogy használja a B$ dobozban található színt, és rajzoljon le madárformájú alakokat vele. A PRINT B$ utasításnak ugyanaz a hatása, mintha begépelnénk egy speciális színvezérlő karaktert. A B$ használatával a VIC-kel választjuk ki automatikusan a színkaraktert. Ahogy az N (2. sor) változik, változik a B$ (3. sor) és a madarak színe.

A VIC véletlenszám generáló képességét összekombinálhatjuk érdekes módon az ő szín-, hang- és grafikus tulajdonságaival. Például újra végezzük el a VIC szín-showt, de véletlenszerűen, és csináljunk egy kis hangot hozzá. Ebben az utolsó példában a VIC színét és hangját véletlenszerűen képezzük.

Gépelje be a következő sorokat a számítógépbe:

```
NEW
1   POKE 36878,3                    (beállítja a hangerőt)
2   C = INT (RND (1) * 255) + 1     (véletlen szám 1—255-ig)
3   S = INT (RND (1) * 50) + 175    (véletlen szám 175 és 224 között)
4   POKE 36879, C                   (beállítja a színt)
5   POKE 36875, S                   (megszólaltat egy hangot)
6   FOR T = 1 TO 100: NEXT T        (várakozás)
7   GOTO 2
```

Amennyiben készen áll a példa futtatására, törölje a képernyőt, és gépelje be a RUN-t, majd nyomja meg a RETURN-t.

Micsoda hang- és színkavalkád! Nem várható el, hogy a véletlenszerű zene mindenkinek tetsszen!

Amennyiben le akarja állítani ezt a szín- és zenegépet, nyomja le a RUN/STOP-ot. Ezután a következő két sort kell hogy begépelje, hogy elhallgattassa a hangot, és a VIC képernyőjét normál színekre visszaállítsa.

```
POKE 36875,0        (elzárja a hangot)
POKE 36879,27       (visszatérnek az eredeti színek)
```

Itt az ideje a saját próbálkozásoknak. Gyakorolja a VIC-színeket. Szórakozzon a színes, muzikális VIC számítógéppel.

BILLENTYŰZET GRAFIKÁK

A VIC 20 speciális grafikus billentyűkészlettel rendelkezik. A billentyűk túlnyomó többségének az előlapján két grafikus karakter látható. Ön ezeket a grafikus karaktereket is kirajzolhatja a képernyőre, vagy ha van Commodore mátrix nyomtatója, akkor kirajzolhatja az összes billentyűzeten található szimbólumot a papírra.

Ha a billentyű bal oldalán látható grafikát akarja kirajzolni, tartsa lenyomva a Commodore billentyűt, és nyomja le a grafikus billentyűket, ha pedig a jobb oldalit akarja, úgy a SHIFT lenyomása mellett üsse le az annak megfelelő billentyűt.

Figyelem: ha a SHIFT és a Commodore billentyűket egyszerre nyomja le, átkapcsol nagy/kisbetűs üzemmódra. A nagy/kisbetűs üzemmódban csak a bal oldali grafikák érhetők el. Ahhoz, hogy visszatérjen a nagybetű/grafika üzemmódba, nyomja meg a SHIFT és Commodore (C=) billentyűket együtt.

A legegyszerűbb módja a VIC grafikák használatának a SHIFT vagy C= és a kiírni kívánt grafikus billentyű együttes lenyomása. Például gépelje be a következőt:

SHIFT S

Láthatja, egy kék szívet írt ki a képernyőre. Próbáljon ki más grafikus billentyűket. Ime néhány billentyű, melyet érdemes kipróbálni.

Megjegyzés

A vonalak különböző méretekben es novekményben jelennek meg, igy pontosan olyan grafikát lehet készíteni, amilyet ön szeretne

BAL OLDALI GRAFIKA JOBB OLDALI GRAFIKA

C= T	SHIFT Q
C= @	SHIFT W
C= +	SHIFT M
C= B	SHIFT N

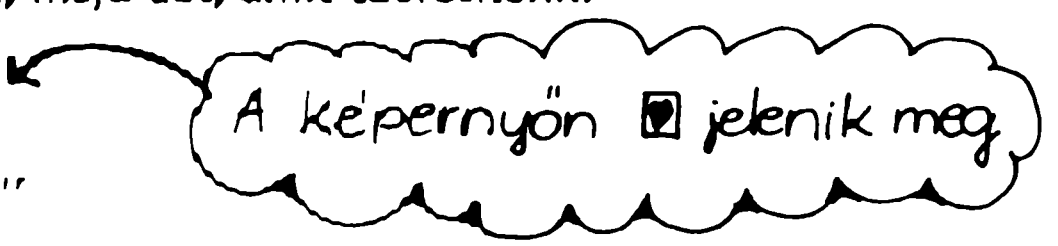
A baloldali grafikák kiválóak táblázatok, formanyomtatványok készítésére. A jobb oldali grafikák kiválóak illusztrációkra, animációra, még kártyajátékra is.

FEJLÉCEK, CÍMEK GRAFIKÁJA

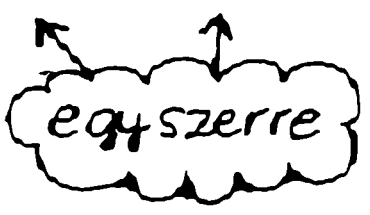
A grafikák nemcsak rajzokra, játékokra alkalmasak. Egy sereg speciális hatás van táblázatok, ábrák címeinek vagy egyes programok fontos részeinek a kiemelésére. A legegyszerűbb mód a szavak, kifejezések kiemelésére, hogy negatívban írjuk ki őket.

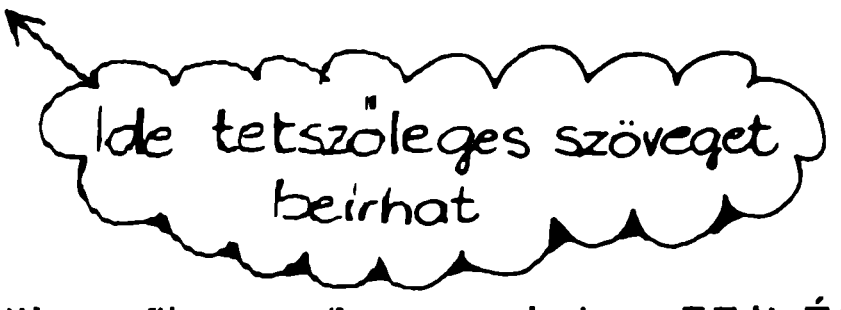
Egyszerűen üssük be a CTRL, RVS/ON-t, majd azt, amit szeretnénk.

Például:




```
5    PRINT " SHIFT  CLR/HOME "
10   PRINT " CTRL  RVS/ON  FEJLÉC"
RUN
```

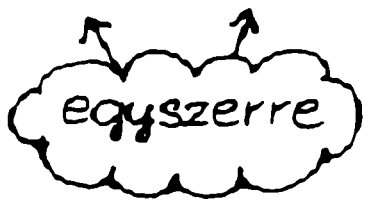




Megjegyzés: A 10. sorban a CTRL RVS/on billentyűket együtt nyomja le, a FEJLÉC-et írja ki.

Most próbáljon ki néhány space-t a címcsíkhoz.

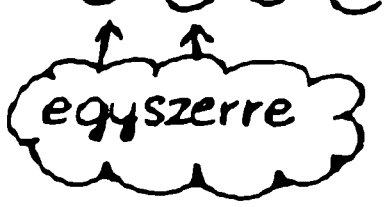
```
10   PRINT" CTRL  RVS/ON  SPACE  CTRL  PUR  FEJLÉC  SPACE "
és   RETURN
RUN
```



Megjegyzés: a 10. sorban a FEJLÉC-en kívül a többi mind egy billentyű. A PUR (bíbor) helyett bármely más szín is választható. A RETURN-ről a sorok végén ne feledkezzen el.

A másik megoldás, hogy egy négyzetet rajzolunk köréjük. A technikája olyan, mint egy rajzolás. Ellenőrizze, hogy pontosan gépeli-e be. Csinálja lépésről lépésre.

```
10   PRINT " C= @ C= @ C= @ C= @ C= @ "
RUN
```

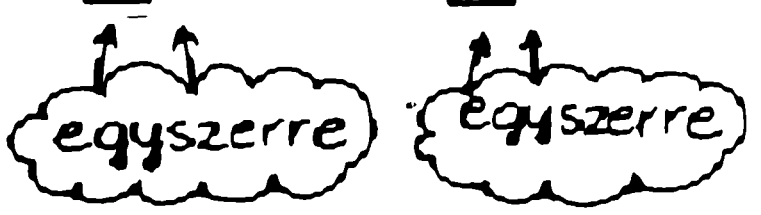


Megjegyzés: Tartsa lenyomva a C= billentyűt, és ötször üsse be a @ gombot.

A képernyőn most egy egyenes vonal lesz. A következő sor a FEJLÉC és mind a két oldalán egy-egy függőleges vonal, hogy meglegyenek a doboz oldalai.

20 PRINT"[C=]GAB C[C=]M"

RUN



Megjegyzés: Ez hosszabb is lehet, de ekkor a dobozt is meg kell hosszabbítani a 10. és 30. sorban.

Látja már, hogy építjük fel a dobozt? Csak be kell fejezni.

30 PRINT"[C=]T[C=]T[C=]T[C=]T[C=]T

RUN

A program editálásához gépelje be, hogy LIST, majd nyomja meg a RETURN-t. Most vissza tud menni és kijavíthatja azt a sort, amelyik nem teljes hibátlan. Használja a CRSR és az INST/DEL billentyűket a hibás rész megközelítéséhez és átírásához. Miután elvégezte a javítást, ahhoz, hogy bevigye a gépbe a változtatást, melyet egy sorban végrehajtott, nyomja le a RETURN-t.

Vagy akár teljesen újraírhatja a sorokat, majd a RETURN-nel lezárhatja őket.

Ime egy másik módszer szavak, fejlécek kiemeléséhez. Ezek villognak néhányszor, amikor a képernyőre kerülnek.

NEW

```
10  FORH = 1 TO 250
20  PRINT "[SHIFT] [CLR/HOME]  FEJLÉC"
25  FOR  T = 1 TO 50: NEXT
30  PRINT "[SHIFT] [CLR/HOME] [CTRL] [RVS/ON]  FEJLÉC"
40  FOR  T = 1 TO 50: NEXT
40  NEXT H
```

RUN

Amennyiben a FEJLÉC mocorog a képernyőn, ezt az idézőjelen belül üres karakterekkel próbálja megszüntetni. A villogás sebességét a 25. és 35. sorban levő szám nagyságával tudja állítani.

4. FEJEZET

ANIMÁCIÓ

Repülő madarak

Ugráló labda

Kurzor ellenőrzés

Gépelje be az alábbi programot pontosan, és nézze meg, mi történik! A program megállításához használja a RUN/STOP gombot.

```
10 PRINT " CLR HOME ";
20 PRINT " ◤ ● ◢ "
30 PRINT " ■ ▒ ■ "
40 PRINT " ◢ ■ ◤ "
50 FOR T = 1 TO 300: NEXT
60 PRINT " CLR HOME ";
70 PRINT " ■ ● ■ "
80 PRINT " ◢ ▒ ◤ "
90 PRINT " ■ ▌ ▐ "
100 FOR T = 1 TO 300: NEXT
110 GOTO 10
```

REPÜLŐ MADARAK

Ez a fejezet bemutatja, hogyan használhatjuk a VIC grafikus képességeit, hogy azt az illúziót keltsük, mintha a karakterek mozognának a képernyőn. A mozgás illúzióját, melyet a VIC képernyőjén keltünk, gyakran életrekeltésnek, animációnak nevezzük. Ez az „életrekeltés" bármilyen programot — legyen az játék stb. — érdekessé, izgalmassá tesz.

Kezdjük a következő sorok begépelésével:

Tartsuk lenyomva a SHIFT billentyűt, és nyomjuk le a CLR/HOME gombot.

NEW és RETURN

1 PRINT "♥ ⌐¬" és RETURN

2 FOR T = 1 TO 150: NEXT T és RETURN . *(várakozás)*

3 PRINT "♥ ⌐¬" és RETURN

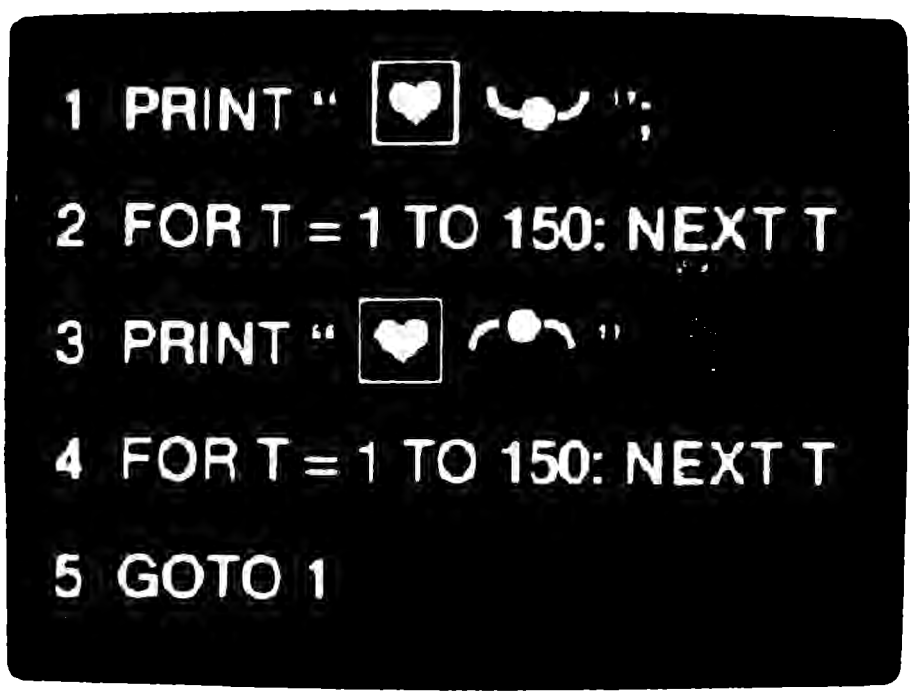
4 FOR T = 1 TO 150: NEXT és RETURN *(várakozás)*

5 GOTO 1 és RETURN

Megjegyzések: a szív ábrát a SHIFT és CLR/HOME együttes lenyomásával kapjuk. Madarak készítéséhez:

LENYOMVA TARTJA	BEÜTI
⌐¬ SHIFT	J Q K
⌐¬ SHIFT	U Q 1

Amit begépelt, a következőképp néz ki:



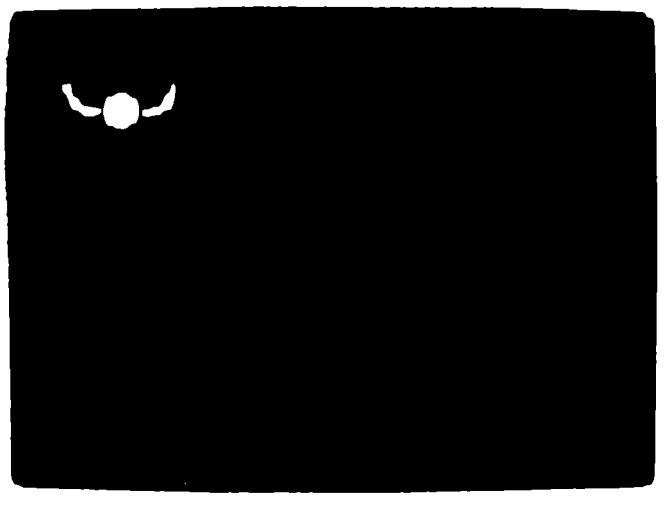
Ellenőrizze, amit begépelt. Megegyezik a látottal? Ha nem, úgy az eltérő sort adja meg újra. Ha minden stimmel, üsse be RUN, és nyomja meg a RETURN billentyűt.

Megjegyzés: ön újra bevihet egy átírhat egy sort. A 2. beírás automatikusan törli és pótolja az előzőt. Egy sort úgy is törölhet, hogy beírja a számát, és megnyomja a RETURN-t.

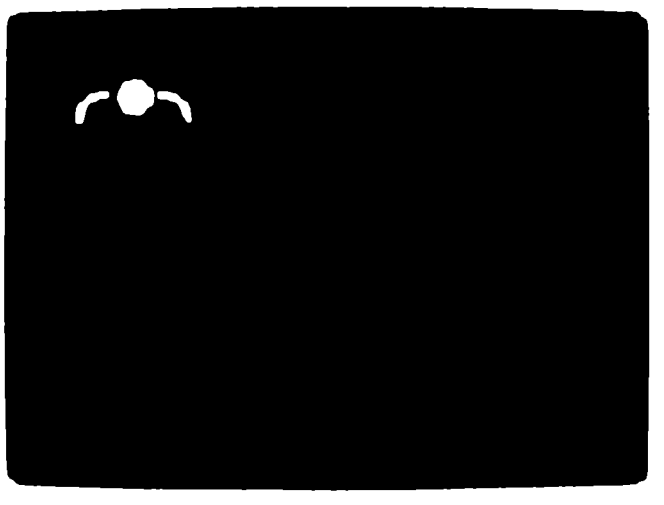
Ime az ön első VIC-animációja. A „madár" verdes a szárnyával a képernyő bal felső sarkán.

Ugy tűnik, erős ellenszél van, mert komoly szárnycsapásokat kell kifejtenie, pedig nem megy sehova. A VIC-életrekeltés titka megtalálható ebben a példában. Hogyan lehet a mozgás illúzióját kelteni? Néhány lépésben megmagyarázható. Ime egy rövid összefoglalás, hogyan lehet bólintani, pislantani, villogni, csóválni, valamilyen mozgás illúzióját kelteni. Először a

karaktert vagy karakterkombinációt írja ki a képernyő valamely pozíciójába.



Másodszor, várjon egy rövid ideig. Üres FOR ... NEXT ciklussal ez megoldható. A repülő madár programban az időkésleltetésre a 2. és 4. sor utasításai mutatnak példát. Ezekben a sorokban a VIC 1-től 150-ig számol. A VIC 150-ig gyorsan elszámol. Próbálja meg a 150-et 500-ra vagy más számra változtatni, és nézze meg, mi történik. Harmadszor, írja ki a karaktert (karakterkombinációt) újra, de egy részét változtassa meg. Az előző késleltetés becsapja önt: elhiteti, hogy „mozgást" látott.



Negyedik: várjon újra. Az elérendő effektustól függően ez a várakozás lehet hosszabb vagy rövidebb az előzőhöz képest. Ebben a példában a késleltetések egyforma hosszúságúak.

Ötödik (egyszerű rész): megismétli az első fent említett 4 részt. Ez egy GOTO utasítással valósítható meg. Megadja a VIC-nek, hogy ugorjon vissza az első sorra, és kezdje az egészet elölről. Így a program lerajzolja a madarat szárnyakkal felfelé, számol 150-ig, majd lerajzolja szárnyakkal lefelé, és visszaugrik az elejére.

Az első és a harmadik lépésnél, amikor a karaktert kinyomtatjuk, gyakran szükséges, hogy meggyőződjünk arról, nem maradt-e régi karakter a képernyőn. A mi madaras példánkban ügyeljünk arra, hogy kiadjuk a képernyő törlő utasítását (SHIFT és CLR/HOME) minden üzenetnél. Ez a karakter törli a képernyőt, a bal felső sarokba állítja a kurzort, és így a madár mindegyik ábrája a bal felső sarokban jelenik meg.

Foglalkozzunk még egy kicsit ezzel a példával.

A RUN/STOP megnyomásával állítsa le a szárny csapkodását, és változtassa meg a számlálás felső határát a 2. és 4. sorban. Fel akar szállni a madara. Próbálja ki az 50 értéket mind a két

esetben. Repüljön a madara gyorsabban? Mit szól, mondjuk, 500 vagy 1000 felső határhoz? Törölje a képernyőt a SHIFT és a CLR/HOME gombok együttes lenyomásával (mindig a SHIFT-nek kell előbb lenyomva lenni). Programon belül is közölheti a VIC-kel, hogy törölje a képernyőt. Ezt a következő begépelésével teheti:

PRINT "♥"

Ez a karakter akkor jelenik meg, ha a SHIFT és a CLR/HOME billentyűt együtt nyomja meg. Ha PRINT utasításon belül megnyomja a SHIFT és CLR/HOME billentyűket, egy negatív szív jelenik meg. Ez a szimbólum jelzés a VIC-nek, hogy törölni kell a képernyőt, és a kurzort a bal felső sarokba (home pozíció) kell vinni.

Próbálja beírni a PRINT utasítást közvetlenül. Törli a képernyőt? Irja be:

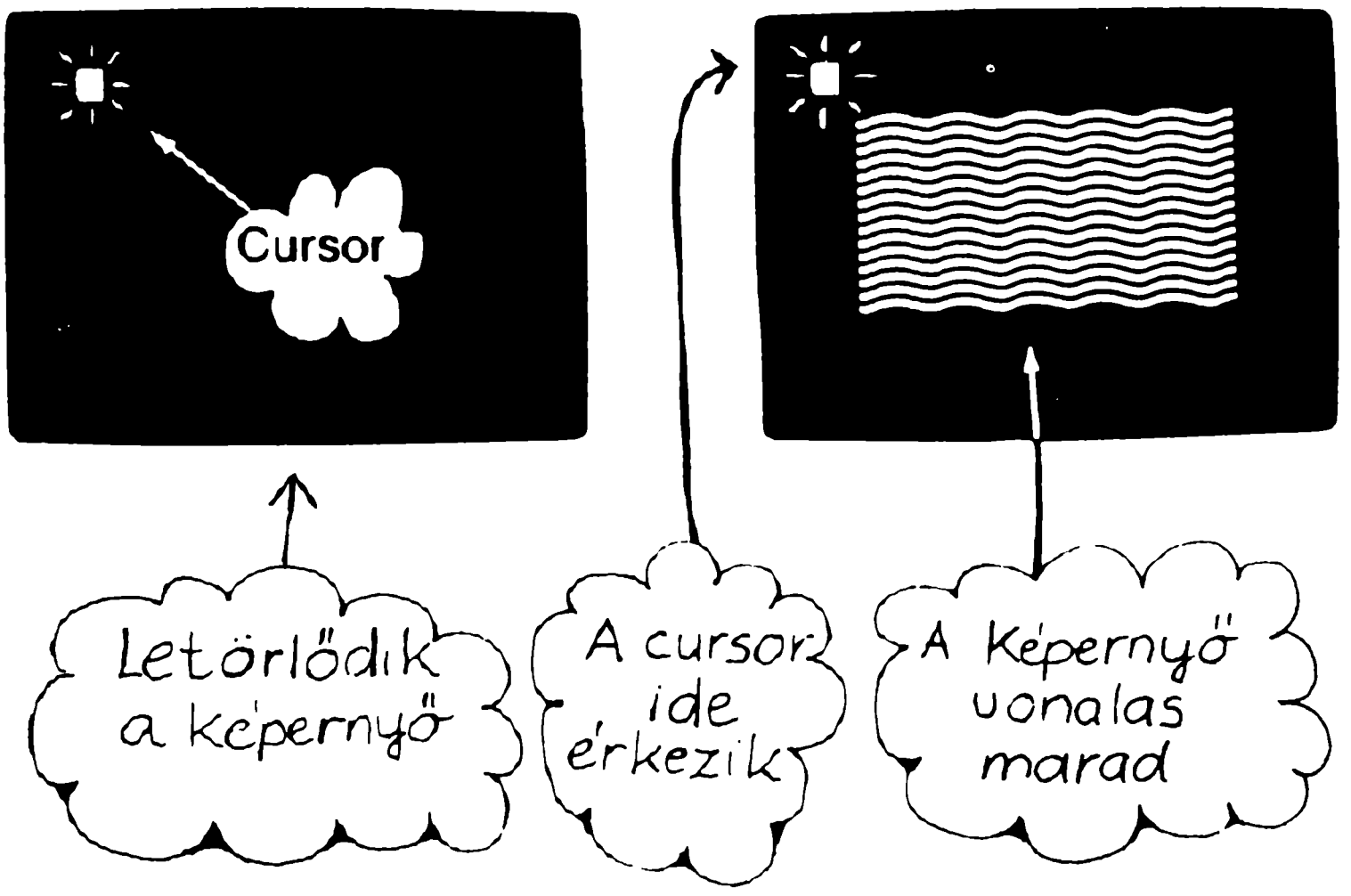
PRINT "♥"

Igen, elvégzi direkt üzemmódban is.

Most írja tele a képernyőt madarakkal, karakterekkel, számokkal, grafikus szimbólumokkal. Miután a képernyő tele van, gépelje be a következő sort:

PRINT "S";

Ezt úgy kapja, hogy SHIFT nélkül nyomja le a CLR/HOME-t. Észrevette a különbséget az előző PRINT utasításhoz képest? A kurzor home pozícióba ment, de a képernyőt nem törölte. Amennyiben nem tartja lenyomva a SHIFT billentyűt, mialatt leüti a CLR/HOME-t, egy negatív S keletkezik a szív helyett. Az S megmondja a VIC-nek, hogy vigye a kurzort a bal sarokba, de ne törölje a képernyőt.



UGRÁLÓ LABDA

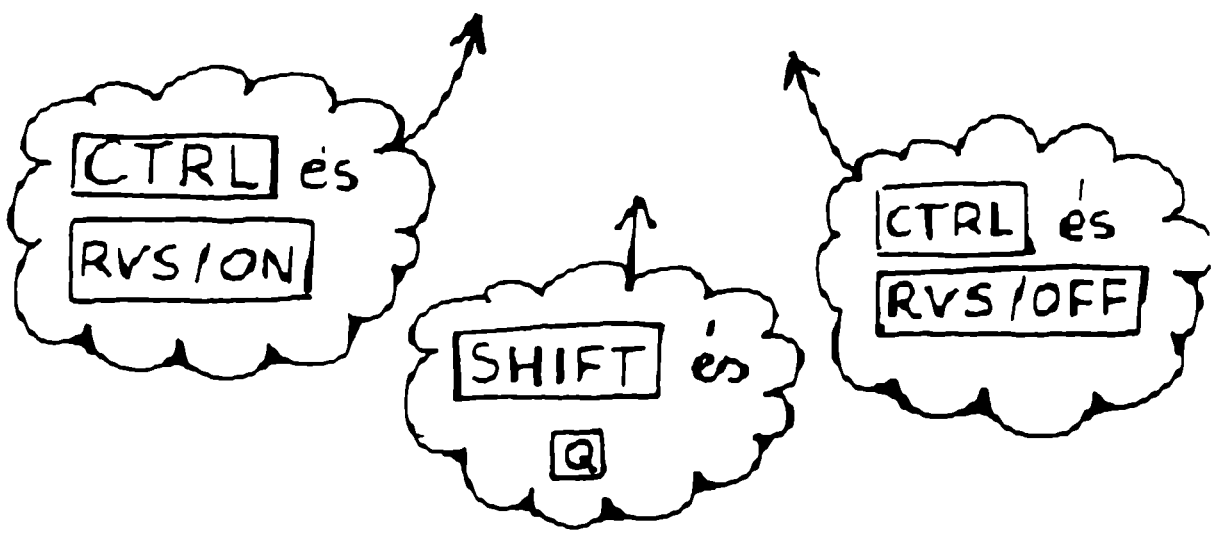
Törölje a képernyőt, és adja be a következőket a VIC-be. Minden sor után nyomja le a RETURN billentyűt.

```
NEW

1     PRINT " ● ";                    (labda = [SHIFT] [Q])

2     GOTO 1

RUN
```

Miután a RUN-t begépelte, a képernyőt feltölti kék labdák oszlopaival.

A RUN/STOP megnyomása BREAK és READY üzenetet eredményez. Most írja be:

```
1     PRINT "[SPACE] [R] [●] [→] ";
```



A két negatív karakter azokat a szimbólumokat jelenti, melyet a VIC kiad, amikor lenyomva tartja CTRL-t és lenyomja az RVS/ON és RVS/OFF billentyűket.

Miután beírta az új sort, futtassa le ezt a példát. Irja be:

```
RUN
```

Mi történik a labdákkal? Nyomja le a CTRL billentyűt, hogy lassítsa a kiírást. Új oszlopok vannak, kék négyszögekből, melyekben fehér lyukak vannak. Sejti, hogy miért?

Az RVS/ON vezérlő karakter a labdaszimbólum előtt azt eredményezi, hogy a VIC „fordított" színeket használ a labda ábrázolásánál. Eredetileg a labda kék, fehér háttérrel. A fordított (negatív) esetben kék háttérben van fehér labda.

A labda utáni RVS/OFF vezérlő karakter közli a VIC-kel, hogy fordítsa vissza a karakter és a háttér színét. Ez olyan, mint a negatívja, és végül oda kerül, ahonnan elindult. Ha nem használja az RVS/OFF-t, érdekes dolog történik.

Állítsa meg (RUN/STOP) a negatív labdákat, és üsse be ezt a sort:

```
1     PRINT " [R] ● ";
```

Az idézőjel után SPACE van, majd RVS/ON és labda

Írja be: RUN

Ha az RVS/OFF-t nem használja, a space (üres karakter) a képernyőn „negatív" lesz, és mint egy tömb jelentkezik. Különféle mintákat lehet így rajzolni! Megtervezheti a padlószőnyeg, tapéta stb. mintázatát.

KURZOR BILLENTYŰK

A négy kurzor-vezérlő billentyűvel bárhova elvihetjük a kurzort a képernyőn. PRINT utasításon belüli kurzor-vezérlők segítenek a kiíratáson belüli karakter pozícionálásánál. Tekintsük a fejezet elején levő „madár" példánt néhány további kurzor-vezérlővel.

Törölje a képernyőt, és üsse be a következő új madár display-t.

```
NEW
1     PRINT "Q J ◄◄ ◄◄◄"
```

Megjegyzések:

Q CRSR le

J CRSR jobbra

◄ CRSR balra — ezt a SHIFT és a RSR együttes lenyomásakor kapjuk.

Ez a sor több kurzor-vezérlőt használ (le, jobbra, balra), mely jobbá teszi a madárképet.

Látni fogja a mozgást, ha lefuttatja az egész példát:

```
2     FOR T = 1 TO 150: NEXT T          (várakozás)
3     PRINT "⌐•¬ ◄◄◄   ";               (madárrajzolás, majd háromszor CRSR balra)
4     FOR T = 1 TO 150: NEXT T          (késleltetés)
5     PRINT "          ";               (3 space, majd 3 CRSR balra)
6     FOR T = 1 TO 150: NEXT T          (várakozás)
7     GOTO 1
```

Gépelje be, majd ellenőrizze a beírtakat. Miután kész van, üsse be a RUN-t, és figyelje a madarat, ahogy a képernyő közepe felé repül.

Ebben a példában a kurzor vezérlő karakterek a madarat lefelé és jobbra mozgatják a képernyőn keresztül. Most pedig ön következik! Készítsen egy másik figurát, és mozgassa azt is a képernyőn!

ÉLETREKELTÉS POKE-KAL PEEK-KEL

Eddig megtanulta a PRINT használatát a kurzor-vezérlőkkel és a szín-vezérlőkkel, amint mozgatják a tárgyat a képernyőn. Valószínűleg a BASIC-et használva ez a leggyorsabb módszer, de nem a legelegánsabb. A játékok általában azt igénylik, hogy a képernyőn mozgó tárgy reagáljon más tárggyal való találkozásra. Egyetlen célravezető, hasznos út ennek a problémának a megoldására a PEEK és POKE utasítások használata.

Ahhoz, hogy használni tudja ezt a technikát, először meg kell értenie a video memóriatérkép koncepcióját. Minden egyes képernyőpozíciónak megfelel egy cím a VIC memórián belül (RAM). Minthogy 506 lehetséges pozíció van a képernyőn karakterek számára, így 506 hely van fenntartva a memóriában a karakterek fogadására. Mivel minden memóriacím 0-tól 255-ig tud számokat tárolni, ezért 256 lehetséges értéke lehet az adott memóriacímnek. Ez az a 256 különböző karakter, melyet a VIC elő tud állítani.

A képernyő-memória a VIC-nél a 7680-as címnél kezdődik és 8185-ig tart. A 7680 cím a bal felső sarok címe. A 7681-es cím a tőle eggyel jobbra levő pozícióé, és így tovább a sorban. A sor utolsó címét követő az első karaktercím a következő sorban lefelé.

Most tegyük fel, hogy vezéreljük az ugráló labdát a képernyőn. A labda a képernyő közepén van a 10. sorban és a 10. oszlopban. (A bal felső sarok a 0. oszlop 0. sor). A képernyő-memóriacímek kiszámítására szolgáló formula:

P = 7680 + X + 22 * Y

ahol X az oszlopot, Y a sort jelöli. Épp ezért a labda memóriapozíciója: 7680 + 10 + 220, azaz 7910

Törölje a képernyőt, és üsse be:

```
POKE 36879,8
```

Ez a képernyő színét feketére változtatja.

Most üsse be:

```
POKE 7910, 81
```

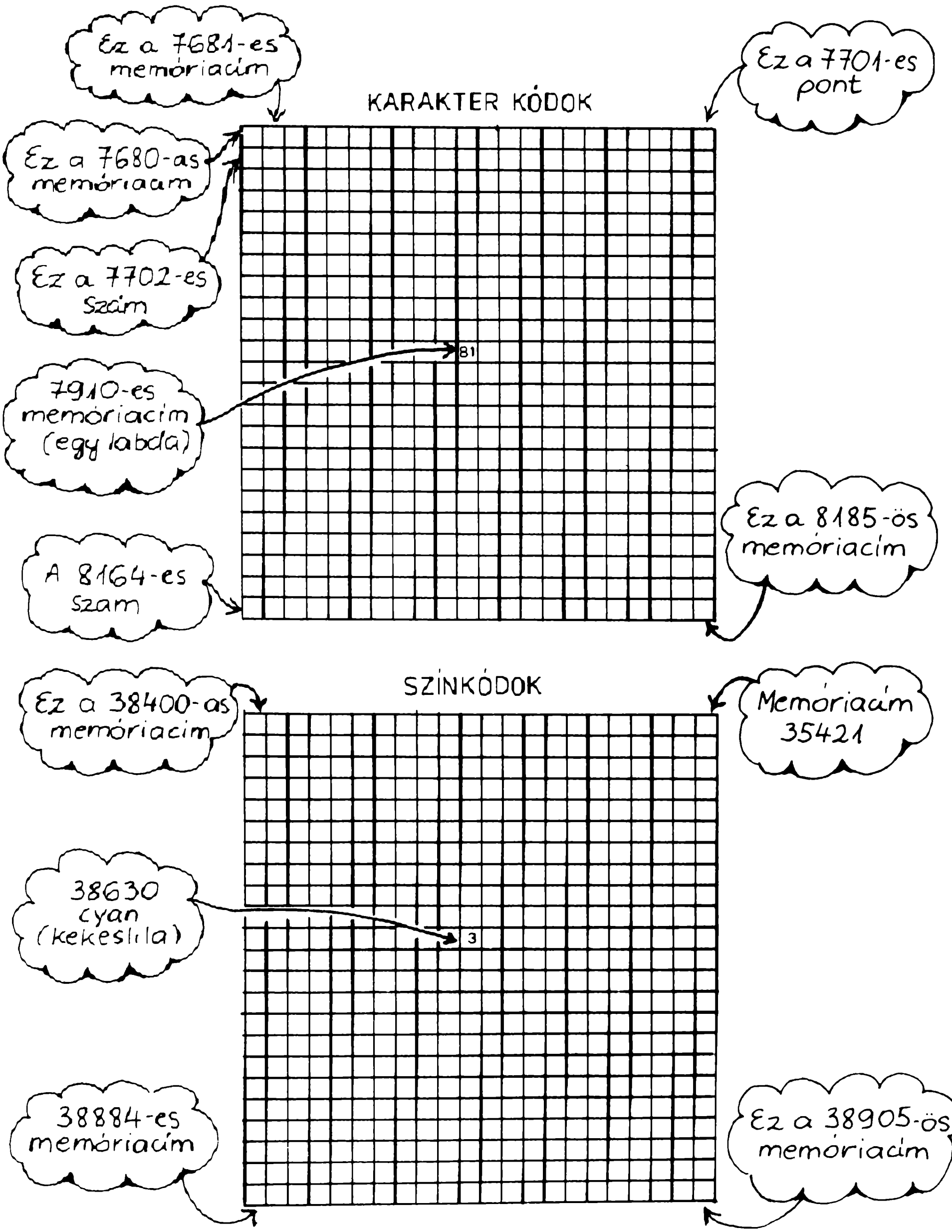
Egy labda jelenik meg a képernyő közepén. Ön most a karaktert közvetlenül a képernyő-memóriába tette a PRINT használata nélkül.

A megjelenő labda fehér. Mindazonáltal lehetőség van arra, hogy a képernyő adott helyén megváltoztassuk a karakter színét a POKE utasítással. Irja be:

```
POKE  38630,3
```

A labda színe világoskékre változik. A VIC képernyőjének minden pontjához 2 memóriacím tartozik: egyik a karaktekód, a másik a színkód számára. A szín memória térkép a 38400-as címnél kezdődik. (Bal felső sarok! és 506 cím hosszban folytatódik. A 0-tól 7-ig tartó színkódok a billentyűzet 1-től 8-ig levő számain találhatóknak felelnek meg.

KÉPERNYŐ MEMÓRIA TÉRKÉP



Íme egy rövid program, mely körbeugráltatja a labdát a képernyőn.

```
10  PRINT "SHIFT CLR/HOME"
20  POKE 36879,9
30  POKE 36878,15
40  X = 1
50  Y = 1
60  DX = 1
70  DY = 1
80  POKE 7680 + X + 22 * Y, 81
90  FOR T = 1 TO 10: NEXT
100 POKE 7680 + X + 22 * Y, 32
110 X = X + DX
120 IFX = 0 OR X = 21 THEN DX = -DX : POKE 36876, 220
130 Y = Y + DY
140 IFY = 0 OR Y = 22 THEN DY - DY : POKE 36 786, 230
150 POKE 36876,0
160 GOTO 80
```

A 10. sor törli a képernyőt, a 20. sor feketére állítja a hátteret fehér kerettel. A 30. sor a hanggenerátort a leghangosabb szintre állítja.

Az X és Y változók követik nyomon a 40. és 50. sorban a labda pillanatnyi oszlop- és sorpozícióját. A DX és DY változók, melyek a 60. és 70. sorban kapnak először értéket, a labda vízszintes és függőleges mozgásának irányát jelentik.

Amennyiben + 1 értéket adunk az X értékhez, az oszlop értéke egyet balra megy. A Y értékhez +1 hozzáadása esetén a labda egy sort lefelé mozog —1 hozzáadása esetén pedig egy sort fölfelé.

A 80. sor jeleníti meg a labdát az aktuális pozíciójában a képernyőn. A 90. sor késleltetéses hurok, mely a szem számára elegendő hosszú ideig tartja a labdát a képernyőn. A 100. sor törli a labdát, mégpedig úgy, hogy a labda helyére üres karaktert (32-es kód) rajzol a képernyőre.

A 110. sor adja az iránytényezőt a X-hez. A 120. sor megvizsgálja, hogy a labda elérte-e már az oldalfalat; megváltoztatja az irányt, és hangot ad az ütközéskor. A 130 és 140-es sorok ugyanezt teszik a fal alján és tetején.

A 150. sor lekapcsolja a hangot, ha azt a program előzőleg bekapcsolta az ütközés érzékeltetése végett. A 160. sor visszaugratja a programot a labda mozgatására, megjelenítésére.

Játsszon ezzel a programmal egy kicsit. Cserélje a számot a 80 sorban 81-ről egy másik karakterkódra, és így a labdát egy másik karakterre. Ha a DX-et vagy a DY-t 0-ra cseréli, a labda átlós irány helyett vízszintes (DY = 0) vagy függőleges (DX = 0) irányban fog mozogni.

Most tegyük egy kissé intellingensebbé a programunkat. Ez idáig csak az X és Y értékeket ellenőriztük, hogy ki ne kerüljön a képernyőről. Bővítsük a következő sorokkal a programunkat. (Egyszerűen írjuk be ezeket a sorokat, és azok automatikusan a programhoz tartoznak.)

```
 32  FOR L = 1 TO 10

 35  POKE 7680 + INT (RND (1) * 506), 102

 37  NEXT

155  IF PEEK (7680 + X + 22 * Y) = 102 THEN DX  DY = -DY
```

POKE 36876, 180 : GOTO 110

A 32–37. sorok 10 szürke karaktert írnak ki a képernyőre véletlenszerűen. A 155. sor (ahol a PEEK van) ellenőrzi, hogy a labda nem a szürke négyzetre akar-e ugrani, és ha igen, megfordítja az irányát.

További információk az értékeléssel kapcsolatban a H) és I) Mellékletben találhatók.

5. FEJEZET

HANG ÉS ZENE

Hangképzés

A VIC négy hangja

Fehér zaj generátor

Dalok játszása

A VIC mint zongora

Néhány szó a POKE-ról

Gépelje be pontosan a programot, ahogy itt látja, és nézze meg, mi történik!

```
NEW
 10  PRINT "[SHIFT] [CRL/HOME]"
 20  FOR I = 1 TO 5
 30  POKE 36873 + I, 0
 40  NEXT I
 50  PRINT "MELYIK HANG (1—4)"
 60  PRINT " HA VÉGE írjon be 0-t"
 70  INPUT N
 80  IF  N = 0 THEN END
 90  PRINT "MILYEN HANGMAGASSÁG (128—254)"
100  INPUT P
110  POKE 36878,4
120  POKE 36873 + N, P
130  FOR J = 1 TO 2000: NEXT N
140  GOTO 10
```

A program megállításához nyomja meg a RUN/STOP billentyűt.

ZENÉLÉS

Ez a fejezet bemutatja azokat a zenei és hanglehetőségeket, melyeket a VIC képes produkálni. Megtanítja önt arra, hogy vezérelje a hangokat, és hogyan játsszon zenét BACH-tól a rockig. Először találjuk ki, hogy lehet egy hangot lejátszani. Tartsa lenyomva a SHIFT billentyűt, és üsse le a CLR/HOME gombot.

NEW	és RETURN
POKE 36878,3	és RETURN
POKE 36875,225	és RETURN

Ennél a pontnál hall egy hangot, ami a TV hangszórójából jön. Ha nem, állítson a TV hangerőszabályozóján. Állítsa be a hangszínt, a hangerőt kellemesre.
Mi most egy közepes C-t játszottunk, egyike a VIC repertoárjának. Ahhoz, hogy elhallgattassa, üsse be a következőt:

POKE 36875,0

És most mézzük meg, hányféle hangot tud a VIC elénekelni.

Usse be: NEW és nyomja meg a RETURN-t.

Igy közli a VIC-kel, hogy ön készen áll egy új program írásához. Ez automatikusan törli az előzőt.

Most üsse be ezt a programot:

```
1    POKE 36878,15                és RETURN
2    FOR I = 128 TO 255           és RETURN
3    POKE 36875 I                 és RETURN
4    FOR D = 1 TO 100             és RETURN
5    NEXT D                       és RETURN
6    NEXT I                       és RETURN
7    POKE 36875,0                 és RETURN
```

Ellenőrizze, hogy a VIC képernyőjén levő program megegyezik-e ezzel. Ha valamelyik hibás, írja újra. Ellenőrizze, hogy a sorok sorszámmal kezdődnek. A VIC automatikusan az új sorral helyettesíti a régit. A program kilistáztatása végett gépelje be:

LIST és nyomja meg a RETURN-t.

Ha minden sorrendben van, gépelje be:

RUN és nyomja meg a RETURN-t.

és hallani fogja mind a 128 hangot, melyet a VIC el tud énekelni. (Ha nem, ellenőrizze a begépelt sorokat). Ha újra hallani akarja a hangokat, üsse be:

RUN újra és RETURN

Most, hogy már hallotta a VIC által produkálható hangokat, adunk egy kis magyarázatot arra, hogy is történik mindez. A VIC-en belül van az ún. Video Interface Chip (innen származik a VIC neve), mely mind a hangot, mind a képet kezeli. Ez biztosítja, hogy a VIC hangokat küldhessen a TV hangszórójához. A hangerőt mind a TV hangerőszabályozójával, mind pedig a VIC billentyűzetéről szabályozhatjuk.

A VIC NÉGY HANGJA

A VIC-nek 4 hangja van — azaz 4 fajta hangon tud énekelni egyszerre. Ugy képzelje el, mint — basszus — tennor — szoprán — zaj. Mindegyik hangnak van egy saját „hangszóró"-száma. Ezt a számot használva a hangszóró bekapcsolható, és a zenei hang, illetve zaj kibocsájtására használhatjuk.

A VIC-nek négy „hangszórója" van:

36874 1. hangszóró — basszus

36875 2. hangszóró — tenor

36876 3. hangszóró — szoprán

36877 4. hangszóró — zaj

Ahhoz, hogy egy hangot játsszon, üsse be a POKE-ot, a beszélő (hangszóró) számát, egy vesszőt és a lejátszandó hang számát.

Például gépelje be:

POKE 36874,128 és nyomja meg a RETURN-t.

Ön most megszólaltatta a 128-as, legmélyebb hangot a 36874-es hangszóróval. Mostantól fogva nevezzük ezt a hangszórót S1-nek (Speaker 1 = hangszóró rövidítése), a másodikat S2-nek, a harmadikat S3-nak, a negyediket S4-nek. Az S2 magasabb hangon énekel, mint az S2. Az S4 pedig zümmögő hangot ad — ez az ún. fehér zaj —, melyet hangeffektusok előállításához használnak.

Egyébként lekapcsolhatja a hangot: "POKE"-oljon 0-t a hangszóróba:

POKE 36874,0 és RETURN

Ez egy kissé komplikáltnak tűnhet, úgyhogy egyszerűsítjük az ön számára. Rengeteg gépelést megspórolhat, ha az alábbi rövid programot használja arra, hogy a beszélő számát egy rövidebb formában adhassa meg. Gépelje be az alábbiakat:

NEW

S1 = 36874 és RETURN

S2 = 36875 és RETURN

S3 = 36876 és RETURN

S4 = 36877 és RETURN

V = 36878 és RETURN

Ez lehetővé teszi, hogy a „hangszórókra" az S1, S2, S3, S4 és a hangerőre V rövidítéssel utaljunk. Mielőtt elindítjuk, ellenőrizzük, hogy minden sort hibátlanul vittünk-e be. Most már készen állunk az egyszerű hanggenerálásra.

POKE V, 10 és RETURN

ezzel állítjuk be a hangerőt.

Így 10-et visszük be a hangerő memória címébe. A hangerő értéke 0 és 15 között változhat. Minél magasabb a szám, annál hangosabban szól.

POKE S1, 195

POKE S2, 215

POKE S3, 231

Gyakorolja a hangképzésnek ezt a módját. Nem sokkal egyszerűbb? Ha le akarja állítani a hangot, "POKE"-oljon 0-t abba a hangszóróba, melyet le akar zárni a következőképp:

POKE S1, 0

POKE S2, 0

POKE S3, 0

Az alább látható zenei hangok táblázata azokat az értékeket tartalmazza, amiket megadhatunk a beszélőnek (hangszórónak), hogy különböző hangokat kapjunk. (Megjegyzés: a 128 alatti számok csendet generálnak) Ezeknek az értékeknek S1 – S3-ba való bevitelével dallamokat játszhatunk. Sajnos a POKE használata lassú és unalmas az embernek, de gyerekjáték a VIC-nek. Ezért tegyük a POKE-t egy programba, és hagyjuk a gépet dolgozni. Legyen, mondjuk ez a „zümmögő darázs repülése".

ZENEI HANGOK TÁBLÁZATA

Hangjegy	Érték	Hangjegy	Érték
C	135	G	215
C#	143	G#	217
D	147	A	219
D#	151	A#	221
E	159	H	223
F	163	C	225
F#	167	C#	227
G	175	D	228
G#	179	D#	229
A	183	E	231
A#	187	F	232
H	191	F#	233
C	195	G	235
C#	199	G#	236
D	201	A	237
D#	203	A#	238
E	207	H	239
F	209	C	240
F#	212	C#	241

A FEHÉR ZAJ GENERÁTOR

A negydik hangnak, S4-nek a száma 36877. Ezt zajnak mondjuk, tulajdonképp ez egy fehér zaj generátor, elsősorban különleges hangeffektusok előállítására használjuk. Legyen ez például a repülőgép hangja.

POKE V, 6
POKE S4, 130 *(ez egy nagyon mély hang)*

Ez egy négymotoros szerkezet.
És mi van a széllel, mely a vitorlázórpülő szárnyainál suhog?

POKE S4, 240

Ne feledkezzen el a hangeróről.

POKE V, 4

vagy bármilyen hangerőt beállíthat, amely a legjobban tetszik.

Most kapcsoljuk le a hangot.

POKE S4, 0

Próbáljuk DATA utasítással pótolni, hogy mást is játsszon. Például egy régi családi kedvencet:

```
300  DATA 225, 360, 225, 360, 225, 240
310  DATA 228, 120, 231, 360, 231, 240
320  DATA 228, 120, 231, 240, 232, 120
330  DATA 235, 720, 240, 360, 235, 360
340  DATA 231, 360, 225, 360, 235, 240
350  DATA 232, 120, 231, 240, 228, 120
360  DATA 225, 480
370  DATA -1
```

Amennyiben jobban szereti a klasszikusokat, próbálja ki az alábbi programot.

Megjegyzés: használjon annyi sort, amennyit akar, ha hosszabb részt kíván játszani, de a program miatt 3 vagy 6 hangjegy legyen soronként.

```
300  DATA 217, 400, 213, 400, 223, 400
310  DATA 227, 200, 234, 200, 230, 400
320  DATA 227, 200, 234, 200, 230, 400
330  DATA 223, 400, 227, 400, 217, 400
340  DATA 213, 600, -1
```

A VIC MINT ZONGORA

Végül íme egy program, mely lehetővé teszi, hogy a VIC billentyűzetét úgy használja, mint egy zongorát.

```
10   REM REGISZTEREK CIMEI
20   S2 = 36875          (a hangregiszterek címeinek rövidítése és a hang
30   V = 36878           kikapcsolása)
40   POKE S2, 0
```

```
100  REM H DUR SKÁLA BEOLVASÁSA
110  FOR N = 1 TO 8                  (a hangjegyek beolvasása a 140-től 160-ik
120  READ A (N)                      sor-ból)
130  NEXT N

140  DATA 223, 227, 229              (POKE értékek a H dúr skálához)
150  DATA 231, 233, 236
160  DATA 238, 239

200  REM ZONGORAJÁTÉK                (Hangerőbeállítás)
210  POKE V, 3

220  GET A$ : IF A$ = " " THEN 220   (Vizsgálat, hogy melyik billentyű lett megnyomva)
230  N = VAL (A$)

240  IF N = 0 OR N = 9 THEN 300      (A program befejeződik a "0" vagy a "9" meg-
                                     nyomására)
250  POKE S2, 0                      (Rövid szünet a hangok között)
260  FOR T = 1 TO 25: NEXT T
270  POKE S2, A(N)                   (Hang-megszólaltatás)
280  GOTO 220                        (Következő hangjegy beolvasásához)

300  REM BEFEJEZÉS
310  POKE S2, 0                      (Hang kikapcsolása)
```

Most, miután begépelte a RUN-t és megnyomta a RETURN-t, a felső számokat tartalmazó sor vezérli a különböző hangokat

1	2	3	4	5	6	7	8
DO	RE	MI	FA	SO	LA	TI	DO

A VIC addig játszik egy hangot, amíg le nem nyom egy másik gombot. Ha befejezte, nyomja le a 0-t vagy a 9-et, és ezzel leáll. Ahhoz, hogy a VIC újra játsszon, újra futtassa (RUN) a programot.

Próbálja ki a következőt:

1 1 5 5 6 6 5

4 4 3 3 2 2 1

5 5 4 4 3 3 2

5 5 4 4 3 3 2

1 1 5 5 6 6 5

4 4 3 3 2 2 1 8

9

(Szegény Beethoven!)

Vagy ezt:

3 3 4 5 5 4 3 2

1 1 2 3 3 2 2

3 3 4 5 5 4 3 2

1 1 2 3 2 1 1

0

DALOK JÁTSZÁSA

A VIC „hangszóróit" és hangjegy-érték táblázatát használva maguk is komponálhatnak dalokat, vagy dalos könyvekből begépelhetnek dallamokat. A következő komponáló program mutatja, hogy lehet ezt csinálni.

Gépelje be a következőket:

```
NEW
10   S2 = 36875
20   V  = 36878

100  REM READ UTASITÁS ÉS LEJÁTSZÁSI CIKLUS
     (A REM azt jelenti, hogy ez a sor nem utasítás, hanem megjegyzés. Angolul: REMark)
110  POKE V, 15                    (Hangerő beállítás)
120  READ P
130  IF P = -1 THEN 200            (Ha vége a dallamnak, a 200-as sortól kezdődnek
                                    a befejezés utasításai)
140  READ D
150  POKE S2, P
160  FOR N = 1 TO D : NEXT N       (A P magasságú hangot D ideig megszólaltatja)
170  POKE S2, 0
180  FOR N = 1 TO 20: NEXT N       (Hallgatni arany)
190  GOTO 120                      (Vegyük elő a következő hangjegyet)
200  REM IDE JÖN, HA A HANGJEGY -1 (REM: ez is megjegyzés)
210  POKE S2, 0                    (Csönd legyen)
220  END                           (Vége)
```

```
300  DATA 225, 1000, 228, 1000, 231, 1000      (225, 228, 231: hangmagasságok)
310  DATA 232, 1000, 235, 1000, 237, 1000      (Időtartam-értékek)
320  DATA 239, 1000, 240, 1000                 (Időtartam-értékek)
330  DATA -1                                   (Itt a vége)
```

Az utasítások első csoportjában beállítjuk azokat a hangszórókat, melyeket használni kívánunk. Jelen esetben az S2-t, a hangerőt, és megadjuk a rövidítéseinket. A következő utasításcsoport a 100. sorban kezdődik. A 100. sor egy megjegyzést (REMARK) tartalmaz, mely megmondja, hogy ebben a szakaszban mit is csinálunk. Ezt huroknak, ciklusnak (loop) nevezzük el, mert itt elolvas, majd lejátszik egy hangot, azután visszaugrik az elejére, és megcsinálja ezt újra egy következő hanggal.
A 110. sor bekapcsolja a hangot.
Most megmondjuk a VIC-nek, hogy hogyan találja ki, melyik hangot játssza.

VIC AJÁNLÁS

A programnak nem kell az 1. sorral kezdődnie, vagy azt 1-es számmal ellátni. Indulhat 10-es számú sorral és 10-esével növekedhet. Ebben az esetben visszaugorhat, és további sorokat szúrhat be közéjük, ha kíván. Például a 11. és 12. számú sorokat beszúrhatja a 10. és a 20. sor közé.

A 120. sor közli a VIC-kel, hogy nézze végig a programot és olvassa be a lejátszandó hangokat (P). Ez az információ egy misztikus utasításban van elrejtve, melyet eddig még nem használtunk. Hasonlóan, a 140-es utasítás közli a VIC-kel, hogy olvasson be információkat — D-nek nevezzük — a hang időtartamára vonatkozóan.

Figyelje meg jól a 130. sort. Ennek a sornak az a feladata, hogy leállítsa a program futását, amikor az az utolsó hangjegyet is beolvasta. Ilyen *„vége a dalnak"* jelzés nélkül a program megpróbálna hangjegyeket beolvasni anélkül, hogy lennének, és így hibajelzést fog kiírni.
A 130. sor azt jelenti, hogy amikor a program beolvassa ezt a jelzést, a "—1" értéket, nem próbálja lejátszani, hanem a program végére ugrik (200. sor). Ne felejtsük el a "—1"-et a mi „misztikus" utasításunk végére odaírni.

Most már tényleg megtanultunk a VIC-kel lejátszani egy hangot, megszüntetni, és visszaugrani egy új hangért.

A 150-es sor betölti a 120. sorban olvasott hangjegyet a hangregiszterbe (2. hang), míg a 160. sorsor a 130.-nak megfelelően késleltetést jelent. Hasonlóan a 170. és 180. sor egy rövid időre elhallgattatja a 2. hangot. A 190. sor hatására a VIC visszaugrik a 120. sorra, hogy beolvassa a következő hangot.

Most már csak a befejező részt kell megírni. Emlékezzünk, hogy a dal végét jelző "—1" a progmot a 200. sorra ugratja.

A 210. sor kikapcsolja a hangot, és a 220. sor azt mondja a VIC-nek, hogy fejezze be a program végrehajtását.

Azzal, hogy megírtuk a befejező részt, még nem vagyunk teljesen kész. Még meg kell írni a „misztikus utasításokat", melyek megadják a programnak a beolvasandó hangjegyeket. Ezt a

„misztikus utasítást" DATA utasításnak hívják, mivel ez tartalmazza a szükséges adatokat. A DATA utasítás a programon belül bárhol lehet. Amikor a program egy READ utasítást végrehajt, keres egy DATA utasítást, ahonnan adatokat beolvashat.

Ez a modul a C dúr skála értékeit tartalmazza. A 300. sor az első három hangjegyet tartalmazza. Az első szám az első betöltendő hangjegy értéke, 225 — a mély C. A második szám mindig az idő, 1000 — kb. 1 másodperc. A 310-es sor a következő három hangjegyet tartalmazza, míg a 320. sor az utolsó kettőt. Magukat az értékeket a zenei hangjegytáblázatból vettük. Amennyiben a sorok hibátlannak tűnnek, lehet futtatni. Önnek egy tiszta C dúr skálát kell hallani. Ha hamisnak tűnik, változtasson a 300. sortól kezdődő DATA utasítás értékein.

Még egyszer: A DATA utasítást programon belül bárhova beteheti. Ezek az értékek sorban kerülnek beolvasásra, a legalacsonyabb sorszámú utasítástól kezdve, az összes DATA utasításon át a végéig.

NÉHÁNY SZÓ A POKE-RÓL

A POKE utasítás lehetővé teszi, hogy a VIC-kel egy egészen más szinten foglalkozzon. A POKE segítségével lehetőség van arra, hogy egyedi memóriacímekhez hozzáférjen, és kicserélje az ott tárolt információt. Mivel ez az utasítás közvetlenül a VIC memóriáján dolgozik, így olyan hibákat is elkövethetünk, mint például a helyes értéket rossz címre vagy rossz értéket jó címre helyezünk. Még egyszer megismételjük, amit már az első fejezetben mondottunk: a billentyűzetről nem lehet kárt tenni a VIC-ben . . . még a POKE-kal sem. De a VIC-et megbolondíthatja. Például, ha már készen van ezzel a résszel, próbálja meg begépelni:

```
POKE 788,0
```

Meglátja, hogy az ilyen módon inzultált géppel csak a RUN/STOP és a RESTORE együttes lenyomása után tud beszédbe elegyedni. Ha túl nagy csapás érte a gépet, akkor kapcsolja ki, majd újra be. Ez nem károsítja a gépet, viszont ön elveszti azt a programot, amin éppen dolgozik. Ha ez megtörténik, akkor az újragépelési idő a veszteség. Mindez azt sugallja, hogy óvatosan bánjunk a POKE utasítással. Ez is kiemeli az egyéb tartozékok szükségességét, mint például a diszk meghajtó, a kazettás egység, melyek lehetővé teszik a programok tartós tárolását, és a nyomtatóét, mely segítségével kinyomtathatjuk a programot. Listáztathatjuk a programunkat, mely az elvesztett program újra történő begépelésekor nagy segítséget nyújthat.

Ajánlatos a C) mellékletben leírtakat a POKE utasítással kapcsolatban tanulmányozni, mielőtt válogatás nélkül elkezdi a POKE utasítást használni. Mindenesetre próbálja ki a könyvben bemutatott mintapéldákat. Óvatosan gépelje be az ott található hosszú számokat, és kétszer is ellenőrizze, mielőtt elkezdené futtatni a programját. Ha rosszul táplálta be az adatokat a gépbe, az könnyen megbolondul, és a hibakeresés során ön is.

6. FEJEZET

TÁRSALGÁS
A VIC SZÁMÍTÓGÉPPEL

Mi az ön neve

Változók bevezetése

Válasszon egy hangot

A GET utasítás

Gépelje be ezt a programot pontosan úgy, ahogy látja, és nézze meg, mi történik!

```
NEW
10    INPUT "HŐMÉRSÉKLET FAHRENHEIT"; F
20    PRINT F "FAHRENHEIT FOK"
30    PRINT "AZ" (F-32) * 5/9 "CELSIUS FOK"
40    PRINT
50    GOTO 10
RUN
```

A 40. sorban levő egyedülálló PRINT utasítás egy üres sort ír ki a képernyőre, amikor fut a program. Próbálja ki a programot a 40. sor nélkül is.

A program megállításához tartsa lenyomva a RUN/STOP billentyűt, és nyomja meg a RESTORE gombot.

Amint látja, az ön VIC-je készen áll, és válaszolni fog a billentyűzeten megadott parancsokra.

Nos, lássunk néhány tréfás dolgot!

Billentyűzze be a következőket: [N] [E] [W], és nyomja meg a RETURN gombot.

Ezután tartsa lenyomva a SHIFT gombot, és nyomja meg a CLR/HOME gombot, majd folytassa az alábbiakkal:

```
1     INPUT
"HOGY SPACE HIVJÁK [SPACE] ÖNT"; A$          és a RETURN gomb
2     PRINT "JÓ [SPACE] NAPOT", A$           és a RETURN gomb
3     GOTO 2                                 és a RETURN gomb
```

Most még négy billentyűt üssön be:

```
RUN                                          és RETURN gomb
```

Ekkor a képernyőn a következő ábrát fogja látni:

```
1     INPUT "HOGY HIVJÁK ÖNT"; A$
2     PRINT "JÓ NAPOT", A$
3     GOTO 2
RUN
HOGY HIVJÁK ÖNT?
```

Figyelje meg, hogy a computer automatikusan kiír egy kérdőjelet, amikor az ön válaszára vár. Ekkor be lehet gépelni a szokásos választ, vagy amennyiben önt VIC-nek hívják, írja:

VIC és nyomja meg a RETURN gombot.

És íme a csoda. Az ön neve tölti fel a képernyőt. Azért, hogy a mozgást lelassítsa, nyomja le a CTRL gombot. Amennyiben megunta a saját nevét nézegetni, nyomja meg a RUN/STOP gombot.

Ahhoz, hogy egy újabb nevet adhasson meg, újra billentyűzze be a RUN-t, és újra megjelenik ugyanaz a "prompt" (súgás), ami a jelen esetben "HOGY HIVJÁK ÖNT? " "Prompt" önnek intézett kérdés, hogy a VIC információhoz jusson. A VIC csak meghatározott módokon kaphat információt a külvilágtól.

A legelterjedtebb módszer az INPUT utasítás segítségével, a billentyűzetről kapott információ. Vizsgáljuk meg részletesen a programunkat. Ime, a következő utasításokat adtuk a VIC-nek.

Első; kiírja a képernyőre "HOGY HIVJÁK ÖNT", majd arra vár, hogy ön, megadja a karaktereket a billentyűzetről. Fogadja a választ, és hozzárendeli az A$-hez. A mi példánkban VIC lesz az A$ értéke. Ily módon gyorsír a computer.

Második; kiírja, hogy "JÓ NAPOT", valamint azt, amit begépeltünk. A mi esetünkben ez VIC.

Harmadik; visszaugrik a 2. sorra.

A második és harmadik lépés, együttesen azt eredményezi, hogy a VIC folytatólagosan kiírja a "JÓ NAPOT, VIC" szöveget a képernyőre. Amennyiben a 3. sort megváltoztatnánk;

```
3     GOTO 1
```

ebben az esetben megjelenne a képernyőn, hogy "JÓ NAPOT, VIC" és rögtön utána: "HOGY HIVJÁK ÖNT". Ennek a változtatásnak eredményeként úgy tűnik, mintha a VIC amnéziában szenvedne.

Amennyiben az INPUT utasítást használjuk, úgy a programunk mindent megszakít addig, ameddig választ nem kap tőlünk. Fontos, hogy a VIC mindaddig vár, amíg nem nyomjuk a RETURN gombot, bármi is történt előtte.

A programunkban egy barátságos "prompt"-ot generáltunk. Mindaddig, ameddig nem közöljük, mit várunk az input üzenetben, addig csak egy egyszerű "? " jelet kapunk. Megpróbálunk olyan "prompt"-ot készíteni, amely megmondja, milyen jellegű inputot vár.

Nyomja meg a RUN/STOP és a RESTORE gombot egyidejűleg.

Gépelje be: NEW, és nyomja meg a RETURN gombot.

```
1     INPUT A$
```

Ha begépeljük a RUN-t, a VIC képernyője a következőképp néz ki:

```
NEW
READY
1     INPUT A$
RUN
?
```

A kérdőjel önmagában több..kérdést vet fel. Próbáljuk ki a következőt:

```
1     PRINT "LENNE OLYAN KEDVES KÖZÖLNI A NEVÉT": INPUT A$
2     PRINT "MI A KEDVENC TORTÁJA": INPUT B$
3     PRINT
4     PRINT "KÖSZÖNJÜK "A$" KEDVES VÁLASZÁT"
5     PRINT "HÓDOLATUNK JELÉÜL HADD KINÁLJUK MEG EGY KIS "B$" TORTÁ-
      VAL"
```

Miután begépelte a RUN-t és megnyomta a RETURN billentyűt, meglátja, hogy a további üzenetek megérik a fáradságot.

VÁLTOZÓK BEVEZETÉSE

A könyvben található programok többsége változókat használ a programok egyszerűsítése, jobb használhatósága érdekében. A változók használata nagyon célszerű és hasznos, hiszen ezeket számok, formulák, grafikus jelek, szavak, kifejezések — akár egész mondatok jelölésére — használhatjuk.

Változatok lehetnek például: X, AB, X$, AB$, SZ$.

A legegyszerűbb mód arra, hogy megértsük ezeknek a változóknak a jelentőségét, az, hogy 255 karakterig képesek valamit képviselni.

Kétfajta változó típust különböztetünk meg: numerikus változók és string változók. A numerikus változókat számok (pillanatnyi számérték) tárolására használjuk.

String változók tetszőleges karakterek (számok, betűk, grafikus jelek, kurzor vezérlés, szín vezérlés stb.) tárolására alkalmasak.

A változókat mint tároló szekrényeket képzelhetjük el a számítógépen belül. Ahhoz, hogy közöljük a VIC-vel, hogy a szekrény számok vagy más értékek tárolására alkalmas, egy speciális nevet használunk. A numerikus változók neve egy vagy két karakter hosszúságú, állhat egy vagy két betűből vagy egy betűből és egy számból. Néhány példa numerikus változóra:

X AB S2 C2 AA ZX

A string változók neve két vagy három karakter hosszúságú (beleértve az $ jelet). Mindig betűvel kell kezdődnie A és Z között, és a dollárjellel $ fejeződik be. Néhány példa string változóra:

X$ AB$ S2$ C2$ AA$ ZX$

Ime egy rövid program, amely bemutatja a változók használatának egy módját.

```
10    A$ = "VIC 20"
20    PRINT "HELLO," A$
RUN
```

A VIC kiírja a képernyőre: HELLO, VIC 20. Miért? Azért mert a 10. sorban megmondtuk a VIC-nek, hogy az A$ változó értéke egyenlő VIC 20-al.

Próbáljuk ki a következőt:

```
10    A = 2
20    B = 3
30    C = 4
40    PRINT A * B * C
RUN
```

Ebben a példában A * B * C egyenlő 2 * 3 * 4, mivel az A, B, C változóknak 2, 3, 4 értéket adtuk.

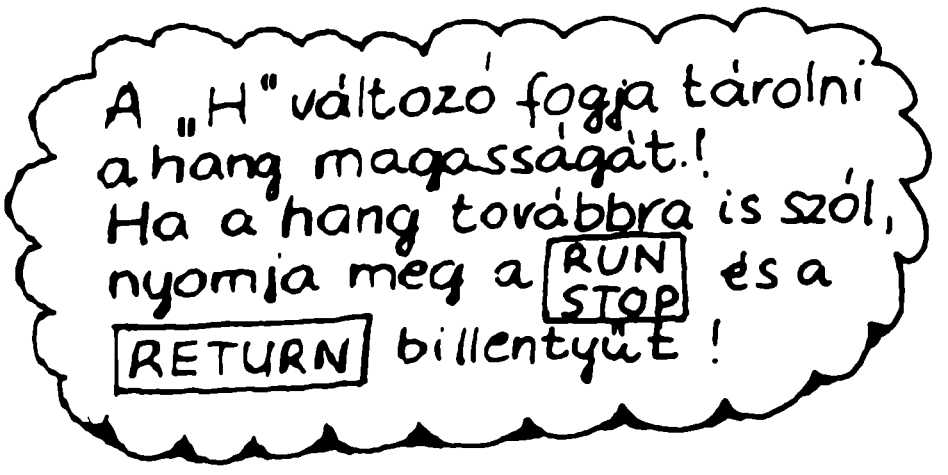
És végül egy utolsó példa, amely mindkét változó típust az INPUT utasításnál bemutatja. Az INPUT utasítás során lehetőség van arra, hogy a program futás közben definiáljuk a változó értéket.

```
10    PRINT "MILYEN SZAVAT KÉPVISEL AZ X$ ": INPUT X$
20    PRINT "MILYEN SZÁMOT KÉPVISEL AZ X":  INPUT X
30    PRINT "AZ X$ ÉRTÉKE MOST" X$
40    PRINT "ÉS AZ X ÉRTÉKE"X
RUN
```

VÁLASSZON EGY HANGOT

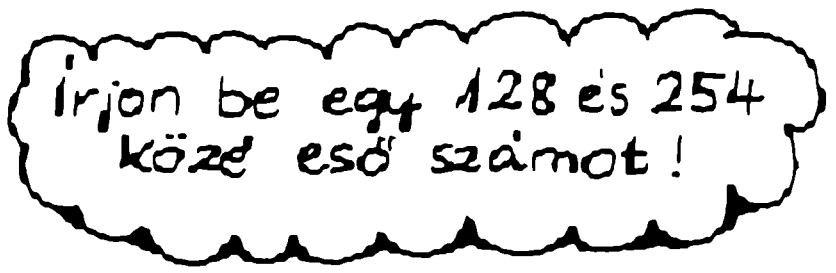
Ezzel az INPUT példával hangot csiholunk ki a VIC-ből. Gépelje be a következőt:

```
NEW
10    INPUT "HANGMAGASSÁG"; H
20    IF H = 0 THEN 90
30    INPUT "IDŐTARTAM"; L
40    POKE 36878,15
50    POKE 36875,H
60    FOR T = 1 TO L: NEXT
70    POKE 36878,0
80    GOTO 10
90    END
RUN
```



A RETURN billentyű megnyomása után a képernyő a következőképp néz ki:

```
RUN
HANGMAGASSÁG ? 225
IDŐTARTAM? 10000
```



Nyomja meg a RETURN gombot, és hallgassa meg a hang magasságát is időtartamát. Minden hang után egy újat generálhat, és így, megismerheti, hogy a misztikus POKE utasítások hogy képezik a hangot. Az első két sor (10. és 30.) biztosítja nekünk azt a rugalmasságot, amellyel lehetőségünk van az 50. és 60. sorban a hangot megváltoztatni. A 20. sor biztosítja a kilépés lehetőségét, amennyiben már eleget próbálkoztunk. (A MILYEN LEGYEN A HANGMAGASSÁG kérdésre 0 (zéró) választ kell adni.) A 70. sor megszünteti a hangot, melyet korábban a 40.-ben aktiváltunk.

A GET UTASÍTÁS

Miután már megismerte az INPUT utasítást, továbblépünk, és bemutatunk egy mulatságosabb módot az adatok billentyűzetről történő beolvasására.

A GET utasítást arra használjuk, hogy karaktereket vigyünk be a gépbe a billentyűzetről, minden alkalommal egyet. Továbbá a program futás során nem szükséges a RETURN billentyűt lenyomni. Lássuk, hogy néz ki ez az utasítás.

```
10    GET A$
```

Hogyan érthető el, hogy a program megálljon és várjon arra, hogy begépeljünk valamit? Beépítünk a programunkba egy IF . . . THEN utasítást a válasz ellenőrzésére.

```
10    GET A$
20    IF A$ " "THEN 10
```

Megjegyzés: Az idézőjelek között nincs üres karakter.

Mit értünk el vele? Ez az egyszerű, kétsoros felhasználói program lehetővé teszi, hogy addig várakozzon a program, ameddig a kezelő le nem üt egy gombot a billentyűzeten. Ez nagyon hasznos, mivel befagyasztjuk vele addig a képernyőt, ameddig a kezelő elolvassa, mit kell csinálni, és eldönti, hogyan tovább.
Bemutatunk egy a GET utasítást példázó progra·not.

```
10    GET A$
20    IFA$ = " "THEN 10
30    IFA$ = "A" THEN PRINT "UJHÁZI TYUKHUSLEVES"
40    IFA$ = "B" THEN PRINT "SPAGETTI"
50    IFA$ = "C" THEN PRINT "SONKA TOJÁSSAL"
60    GO TO 10

RUN
```

Miután a programot begépeltük, és beírtuk a RUN-t, mindaddig várni fog, ameddig az operátor le nem üt egy billentyűt. Amennyiben az A betűt ütötte le, úgy az UJHÁZI TYUKHUSLEVES jelenik meg a képernyőn. A B hatására a SPAGETTI-t, a C hatására pedig a SONKA TOJÁSSAL-t írja ki. Igy egy billentyű lenyomására a VIC egy teljes szót ír ki.

Most bemutatunk egy hosszabb programot, mely a PRINT és GET utasítások célszerű felhasználásait mutatja be egy számítógépes recepten belül. Nem kell megijedni a program hosszától. Nagyon egyszerű PRINT utasításokat használ. A cél az, hogy bemutassuk, hogyan lehet az A$ string változót használni különböző hosszúságú kifejezésekhez, valamint a GET hatását.

```
10 PRINT "■VALASSZON AZ ETLAPROL"
20 PRINT
30 PRINT "A...UJHAZI TYUKHUSLEVES"
40 PRINT "B...SPAGETTI"
50 PRINT "C...SONKA TOJASSAL"
200 GETA$:IF A$=""THEN 200
210 IF A$="A"THEN 500
220 IF A$="B"THEN 700
230 IF A$="C"THEN 300
450 GOTO 200
502 PRINT"■UJHAZI TYUKHÚSLEVES"
520 PRINT
530 PRINT"FOGJ EGY TYUKOT, OLD MEG"
540 PRINT"KOPASZD MEG"
550 PRINT"VEGYEL EGY 4 LITERES FAZEKAT"
560 PRINT"TEDD TELE VIZZEL"
570 PRINT"DOBD BELE A TYUKOT"
580 PRINT"2 ORAN BELUL"
590 PRINT"CSODAS ILLATOK ARADNAK"
595 PRINT
600 PRINT"NYOMJA LE BARMELYIK GOMBOT"
610 GET A$:IF A$=""THEN 610
620 GOTO 10
700 PRINT"■MAMA FELE SPAGETTI"
705 PRINT
710 PRINT"VEGY 1 CSOMAG SPAGETTI TESZTAT"
720 PRINT"VAGJ FEL APRORA 1/2 KG"
730 PRINT"MARHAHUST"
740 PRINT"TEGYEL HOZZA 1 HAGYMAT"
750 PRINT"2DB PARADICSOMOT"
760 PRINT"FUSZEREZD BORSSAL PAPRIKAVAL"
770 PRINT"TEGY HOZZA EGY TOJAST"
780 PRINT"FOZD 1/2 ORAT"
790 PRINT"TEDD TANYERRA"
800 PRINT"SZORJ RA SAJTOT"
810 PRINT"A TALALASNAL NE FELEDKEZZ EL"
820 PRINT"A SORROL"
830 GOTO 590
900 PRINT"■SONKA TOJASSAL"
905 PRINT
910 PRINT"FOZZ MEG EGY SZELET SONKAT"
920 PRINT"FOZZ MEG EGY TOJAST"
930 PRINT"TALALD EGYUTT"
940 PRINT"+ITAL"
950 GOTO530
```


READY.

Amennyiben helyesen gépelte be a programot, a képernyőn a RUN begépelése után a következő látható:

VÁLASSZON AZ ÉTLAPRÓL

A . . . UJHÁZI TYUKHUSLEVES

B . . . SPAGETTI

C . . . SONKA TOJÁSSAL

Ezután a VIC arra vár, hogy a kezelő lenyomjon egy gombot. Amennyiben nem az A, B vagy C gombot üti le, nem történik semmi. (200. sor eredménye.) Ha az A gombot üti le, úgy az Ujházi tyúkhúsleves receptjét kapja meg.

Természetesen a program tetszésünk szerint hosszabbítható, változtatható. Lehet az étlapot bővíteni az 50. sor utáni PRINT utasítás segítségével, valamint a 230. sor utáni új IF utasítás figyelembevételével, mindaddig, ameddig hely van. A recept utolsó sorában a GOTO 590 kell hogy álljon, mely a programot futtató személlyel közli, hogy le kell nyomnia egy billentyűt a folytatáshoz. A receptet annyi ideig tartja a képernyőn, ameddig akarjuk.

Természetesen a most bemutatott programot nem csak receptre lehet használni. Mi van például a név és cím file-okkal?

Az étlap helyett használjuk az emberek neveinek a rövidítését, és a recept helyett azok teljes nevét, címét, telefonszámát. Van már elképzelése a GET, INPUT használatának lehetőségeiről? Ez egy igazi számítástechnikai segédeszköz, melyet a gép az ön szolgálatába állít.

7. FEJEZET

BEVEZETÉS A PROGRAMOZÁSBA

Első BASIC utasítások

Véletlen számok

Gépelje be ezt a programot pontosan úgy, ahogy látja, és nézze meg, mi történik!

```
10  PRINT " SHIFT  CLR HOME  "
20  PRINT CHR$(205.5 + RND(1));
30  GOTO 20
```

A program megállításához nyomja meg a RUN/STOP billentyűt.

AZ ÖN ELSŐ BASIC PROGRAMJAI ÉS AZOK MŰKÖDÉSE

Mind ez idáig ön csupán begépelte a programokat anélkül, hogy értené működésüket. Ebben a fejezetben elmagyarázzuk ezeknek a rafinált kis programoknak a működését, valamint útbaigazítjuk, hogyan programozza a saját VIC gépét.

PROGRAM 1: Az ÖN NEVE villogva

```
10  PRINT " SHIFT  CLR HOME  "
20  FOR T = 1 TO 300 : NEXT
30  PRINT "ÖN NEVE"
40  FOR T = 1 TO 300 NEXT
50  GOTO 10
```

Nem tűnik túlságosan komolynak a program, de ha egyszer ezt megértette, hogy működik, megvan a kulcs ahhoz, hogy ön is hasonlókat készítsen. A 10. sor egy PRINT utasítás, az idézőjelek között olyan karakterekkel, melyek jelentése a képernyő törlése. Amennyiben megpróbálná ezt idézőjelek nélkül begépelni, úgy a programsor eltűnne, mielőtt befejezné a gépelést.

A VIC csak akkor ismer fel egy új sort, miután leütötte a RETURN billentyűt, úgy, hogy a kurzor abban a sorban van.

VIC AJÁNLÁS: idézőjelek

Beszéljünk egy keveset arról, mi történik akkor, amikor leütjük az idézőjel billentyűt.

Elsőként amikor leüti az idézőjelet valami furcsa történik. Ha leüti a HOME, CLR, kurzor fel, le, jobbra vagy balra billentyűket, fordított grafikus karaktert kap. Ugyanez történik, amennyiben akármelyik szín vezérlő billentyűt üti le. Látja az idézőjelet, a programban használjuk is, a VIC úgy tekinti, mint program utasítást. Tehát amikor leüt egy kurzor vagy szín vezérlő billentyűt az idézőjel után a VIC leír a képernyőre egy speciális karaktert, amely jelzi ezt a műveletet. Például, ha begépeli a " jelet, majd lenyomja a SHIFT, CLR/HOME billentyűket, egy negatív szívet kap a képernyőn. Amennyiben

ezt látja egy programon belül, tudni fogja, hogy ez azt jelenti, hogy törli a képernyőt. Más szimbólumok más műveletek elvégzésére vannak fenntartva. Miután másodszor is leütötte az idézőjel billentyűt, az összes kurzor és szín kontrol normálisan fog működni.

Az INSERT billentyűnek ugyanaz a hatása. Amennyiben ezt a billentyűt leütjük, minden, a sorban generált üres karakter úgy viselkedik, mintha idézőjel módban lenne. Az összes kurzor kontrol úgy viselkedik, mintha idézőjel között volna. Továbbá a DELETE billentyű egy speciális negatív karaktert eredményez ezeken az üres helyeken, melyeknek az a hatása, hogy a program listázásakor kitöröl karaktereket és kiír DELETE-ket a képernyőre, amikor nyomtatunk.

A program 20. sorát időkésleltető ciklusnak nevezzük. Ez gyakorlatilag két BASIC utasítás egy sorban, melyeket kettősponttal választunk el. Mi történik itt? Mindössze annyi, hogy a VIC elszámol 300-ig, miközben nem csinál semmit. Ez arra szolgál, hogy egy kissé lelassítsuk a program futását.

(Töröljük ki a 20. és 40. sort, és futtassuk le a programot. Túl gyorsan villog.)

A 30. sorban begépelte a nevét (vagy legalábbis reméljük, hogy megtette). Ennek hatására kiírta az ön nevét a képernyőre. Semmi egyéb nincs a képernyőn, mivel minden mást a 10. sorban töröltünk.

A 40. sorban ismét időkésleltetés történik, hogy ideje legyen elolvasni a nevét a képernyőn.

Az 50. sorban a 10. sorra való ugrás utasítása áll, amely így a következő utasítás lesz, mely törli a képernyőt. Miután megjelent az ön neve a TV-képernyőn, 1 másodperc eltelte után törli. Ujabb egy másodperc múltán megjelenik a név ugyanabban a pozícióban. Majd újra törlődik és kiíródik. Mivel minden alkalommal ugyanabban a pozícióban jelenik meg a képernyőn, azt az érzést kelti a szemben, mintha villogna.

Próbálja ki a programot. A megjelenés és törlés közötti időtartamot növelheti vagy csökkentheti, amennyiben a 20. és 40. sorban megváltoztatja a 300-as számértéket.

PROGRAM 2: Rengeteg szív (3. fejezet)

```
10   FOR H = 1 TO 505
20   PRINT "♥";
30   NEXT
40   FOR C = 8 TO 255 STEP 17
50   POKE 36879, C
60   FOR T = 1 TO 500: NEXT
70   NEXT
80   GOTO 40
```

A program megtölti a képernyőt színes szívekkel. Bemutatja az írásjelek használatát a PRINT utasításon belül, valamint a POKE használatát arra, hogyan kell az alapszínt, a karakter színét változtatni.

A 10. sorban létrehozunk egy ciklust, amely 505-ig számol. 505 szívet akarunk a képernyőn megjeleníteni, mivel 506 hely van a képernyőn. Amennyiben kiíratjuk az 506. karaktert, úgy a képernyő egy sort emel. Ennek következtében kevesebb szív lesz a képernyőn, mint korábban volt.

A 20. sor kirajzolja a képernyőre a szív karaktert. A pontosvesszőnek (;) az utolsó idézőjel után fontos szerepe van. Mint tudja; a normál PRINT utasítás után a VIC még további műveletet hajt végre. A kurzor visszamegy a sor elejére (kocsi vissza), majd a következő sorra ugrik (soremelés). Az írásjel a sor végén megszünteti kocsi vissza, soremelés utasítást, így a következő karaktert közvetlenül az előző mellé nyomtatja.

A 30. sorban lezárjuk a ciklust. Mindaddig, ameddig a H értéke kisebb, mint 505, a program kiírja a szíveket a képernyőre. Miután kiírta az 505.-ket, a program az ezt követő sorral folytatódik (40. sor).

A 40. sor létrehoz egy újabb ciklust, és indítja az 50. sort, amelyik cseréli az alapszínt és a keret színét. C-t számsorozatként definiáltuk 8-tól 255-ig 17-es növekménnyel. Minden alkalommal, amikor a 70. sornál jár (NEXT), 17-et ad hozzá az előző C értékéhez, és az összeg lesz C új értéke. Ez a színek egy ciklusát eredményezi, beleértve a fekete alapszínt és a fekete keret színtől a fehér alapszínt fehér keret színig stb. mind a 8 keret színnel. Mikor a C értéke meghaladta ezen színek értékét, 7 különböző alapszínt választ (ld. az I. mellékletben a színekhez tartozó számok listáját).

Az 50. sor a színváltoztatást végrehajtja. A C változó aktuális értékét a 36879 memóriahelyen tárolja. Ez a VIC-chipnek egy tényleges tárolóhelyén található, nem pedig a mi normál tárolóhelyünkön.

A 60. sor egy késleltető ciklus. Amennyiben elhagyjuk ezt a sort, a sorok olyan gyorsan változnak, hogy megfájdul a fejünk tőle.

A 70. sor lezárja azt a ciklust, amelyet a 40. sorban indítottunk. Vegyük észre, hogy ebben a sorban

```
70  NEXT C
```

is állhatna, de mi nem ezt az utat választottuk.

A 80. sor hatására újrakezdi az egész ciklust a színekkel. Mindaddig fut, amíg meg nem nyomjuk a STOP gombot, vagy ki nem kapcsoljuk a VIC-et. Gépelje be a POKE 36879,27-t miután leütötte a STOP gombot, és a képernyőre visszaáll az eredeti állapot.

```
10 PRINT " CLR HOME ";
20 PRINT " ▚ ● ▞ "
30 PRINT " □ ▒ □ "
40 PRINT " ▞ □ ▚ "
50 FOR T = 1 TO 300: NEXT
60 PRINT " CLR HOME ";
70 PRINT " □ ● □ "
80 PRINT " ▞ ▒ ▚ "
90 PRINT " □ ▌ ▌ "
100 FOR T = 1 TO 300: NEXT
110 GOTO 10
```

A program hasonló az általunk elsőként készített programhoz, melyet az "Ön neve villogva" névre kereszteltünk. Most azonban ahelyett, hogy rajzolnánk egy képmást, majd kitörölnénk, mint ahogy az első programban tettük, most rajzolunk egy komplett képet, majd szünetet tartunk, és az egészet egy másik képpel pótoljuk. A feje és a teste a VIC személynek ugyanabban a pozícióban marad, csak a keze és a lába változik. Ez azt az illúziót adja, mintha egyik pozícióból a másikba menne.

A 10. és 60. sor viszi a kurzort a bal felső sarokba, melyet home pozíciónak hívunk. Ezért lesz a VIC személy alakja mindig ugyanabban a pozícióban a képernyőn.

A 20., 30. és 40. sorok rajzolják meg a VIC személy első képét.

Az 50. és 100. sorok időkésleltetési ciklusok, melyek kellő ideig tartják a képet a képernyőn.

A 70., 80. és 90. sorok rajzolják ki a második formát a képernyőre. Ez olyan gyorsan megy végbe, hogy ön nem is látja az átmenetet, a szemében összefolyik a kép.

PROGRAM 4: Válassz egy hangot (5. fejezet)

```
NEW
 10  PRINT " [SHIFT] [CLR HOME] "
 20  FOR I = 1 TO 5
 30  POKE 36873 + I, 0
 40  NEXT I
 50  PRINT "MELYIK HANG (1—4)? "
 60  HA BEFEJEZTE, UGY 0"
 70  INPUT N
 80  IF N = 0 THEN END
 90  PRINT "MILYEN MAGAS (128—254)? "
100  INPUT P
110  POKE 36878,4
120  POKE 36873 + N, P
130  FOR J = 1 TO 2000: NEXT N
140  GOTO 10
```

Magyarázat a programhoz:

A 10-től 40-ig található sorok letörlik a képernyőt és kikapcsolnak minden hangot.

Az 50-től 70-ig található sorok segítségével kiválaszthat egy hangot.

A 80. sorszámon fejeződik be a program.

A 90. és 100. sorok segítségével adhatja meg a hangmagasságot.

A 110-től 130-ig terjedő sorok állítják be a hangerősséget, a -magasságot és a 2 másodperces késleltetést.

A 140-ik sor hatására az egész folyamat újra kezdődik a 10-es sortól.

Miután ellenőrizte, hogy minden sor hibátlan, próbálja meg futtatni a programot (írja be a RUN-t, és nyomja meg a RETURN billentyűt). Ez lehetővé teszi, hogy kiválasszon egy hangot és hangmagasságot, és a választott hangot 2 másodpercig lejátssza. Miután a hang véget ér, a program újabb hang után kérdez. Ez a program zenei élmény, biztosan megéri kipróbálni. Amennyiben le akarja állítani a programot, a 0-ás hangot válassza. Valószínűleg hibát gyanít, amikor a 3-as hangnál 254-et választ hangmagasságnak, és nem hall semmit. Valójában ez nem hiba — ez a hang túl magas ahhoz, hogy az emberi fül érzékelje (próbálja ki a kutyáján).

PROGRAM 5: Hőmérséklet **konverzió** (6. fejezet)

```
10    INPUT "FOK FAHRENHEITBEN"; F
20    PRINT F "FOK FAHRENHEIT"
30    PRINT "AZ" (F-32) * 5/9 "FOK CELSIUS"
40    PRINT
50    GOTO 10
```

Ez a program bemutatja az INPUT utasítást, mely lehetővé teszi a program megállítását, mely ezután a kívánt információ megadását kéri az operátortól (az a személy, aki a programot futtatja). A 10. sor hatására megjelenik a kérdés a képernyőn: FOK FAHRENHEITBEN? Az INPUT utasítás keretén belül az idézőjelek között levő szavaknak ugyanaz a hatásuk, mint a PRINT utasításnál. Mindazonáltal az utolsó szót mindig kérdőjel követi, és a program itt további információk megadására vár.

A 20. sor kiírja az F értékét, melyet éppen akkor gépeltünk be.

A 30. sor kiírja az átszámítás eredményét. A 40. sorban levő PRINT utasítás hatására egy üres sor jelenik meg a képernyőn.

Végül az 50. sor visszaugratja a programot az elejére, és újabb információt kér az újraindításhoz. Az eredeti kérdést teszi fel újra, és így több hőmérséklet-átszámítás végezhető el. Mikor befejezte, nyomja le egyidejűleg a STOP és a RESTORE billentyűt. Nincs más mód, hogy közölje a programmal, hogy készen van.

PROGRAM 6: Labirintus

```
10    PRINT " [SHIFT] [CLR HOME] "
20    PRINT CHR$ (205.5 + RND (1));
30    GOTO 20
```

Ez a kis program egy ál-labirintust hoz létre a képernyőn. Mint ehogy várható, a 20. sor a program lelke. A CHR$ függvény egy karaktert ad, melynek kódszáma 0 és 255 között van. Minden karakter, amelyet a VIC kiír a képernyőre, kódolható (ld. H) Melléklet).

Ahhoz, hogy megtudjuk egy karakter kódját, gépelje le a PRINT ASC ("X") utasítást, ahol az X az ellenőrző karakter (ne feledkezzünk el az idézőjelekről). Ezután gépeljük be a PRINT CHR$ (X) utasítást. Itt most X a korábban a VIC-től kapott számot jelenti (itt nincs idézőjel). Nézze meg, hogy működik.

Most gépelje be: PRINT CHR$ (205); CHR$ (206). Ez az M és N billentyűk jobb oldalán található grafikus karaktert írja ki. Ez az a két karakter, amelyet a program a labirintushoz felhasznál.

A 205.5 + RND(1) formula hatására a VIC véletlen számot generál 205.5 és 206.5 között. 50 — 50 százalék az esélye annak, hogy a kapott szám 206 fölött vagy alatt van. A CHR$ függvény nem veszi figyelembe a tizedes pont utáni értéket. Épp ezért a kiírt karakterek közel felének a kódja 205, a másik felének pedig 206.

Amennyiben kísérletezgetni kíván a programmal, változtassa meg a 205,5 értékét úgy, hogy néhány tizedet adjon hozzá, vagy vegyen el belőle. Így az egyik karakter előfordulási gyakorisága nagyobb lesz, mint a másiké.

NÉHÁNY SZÓ A VÉLETLEN SZÁMOKRÓL

A véletlen szám generátor egyike a leghasználatosabb és legeredményesebb vonása a BASIC-nek, mely biztosítja különböző típusú játékok elkészítési lehetőségét.

Az X = RND (1) utasítás hatására a VIC kiválaszt egy véletlen számot 0 és 1 között, de azokat nem, és beírja X-be. Ennek eredményeképp X a következő értékeket veheti fel:

$0 < X < 1$

Amennyiben véletlen számokkal kíván dolgozni, a legjobb azt észben tartani, hogy az ember egy tartományon belül generálja azokat, és azt kell az egész tartományra átszámítani. Például, ha a lehetséges értékek 0 és 3 között vannak, úgy az X értékét 3-mal kell szorozni.

Az új tartomány $0 < X < 3$

Amennyiben 10 és 20 között akarunk számokat választani, hogyan kell átalakítani a számítást, hogy a kívánt tartományba kerüljünk? Először 10-et hozzáadunk a kapott számhoz, és így megváltozott a tartomány a következőképp:

$10 < X < 11$

Mielőtt hozzáadnánk 10-et, szorozzuk meg a véletlen számot 10-zel, és így az új tartomány:

$10 < X < 20$

Így a véletlen szám képlete 10 és 20 között a következő:

X = RND (1) * 10 + 10

Ez idáig megtanultuk, hogy lehet a kívánt véletlen szám tartományát meghatározni. Mindazonáltal a függvény eredményeként olyan tizedes helyiértékek keletkezhetnek, melyek nem kívánatosak mindig, mint például a kockajátéknál, vagy pedig amikor 1 és 10 között kell egész számot kiválasztani. Épp ezért bevezetjük az INT függvény használatát. Ez levágja a szám tizedes értékeit. A 10 és 20 közötti véletlen számok képlete az INT felhasználásával:

X = INT (RND (1) * 10 + 10)

Ekkor a következő tartományt kapjuk:

$10 <= X <= 19$

De megálljunk: A felső határ 20-ról 19-re esett vissza. Miért? Mert a tartomány mindig kisebb volt, mint 20. Az INT függvény a tizedesértékeket elhagyja a 19 és 20 közötti számoknál, és 19-et ad eredményül. A tartomány másik végénél a 10 és 11 közötti számok 10-es értékre lesz-

nek lefaragva. Amennyiben továbbra is ragaszkodunk a 10 és 20 közötti számokhoz, a képlet a következőképp alakul:

X = INT (RND (1) * 11 + 10)

A véletlen számot szorozzuk a tartomány nyújtása végett, és hozzáadunk az eltolás miatt.

Egy tetszőleges tartományú véletlen szám általános képlete a következő:

X = INT (RND (1) * a) + b

ahol *a* jelzi a lehetséges értékek számát, és *b* a tartomány alsó határa.

8. FEJEZET

MELLÉKLETEK

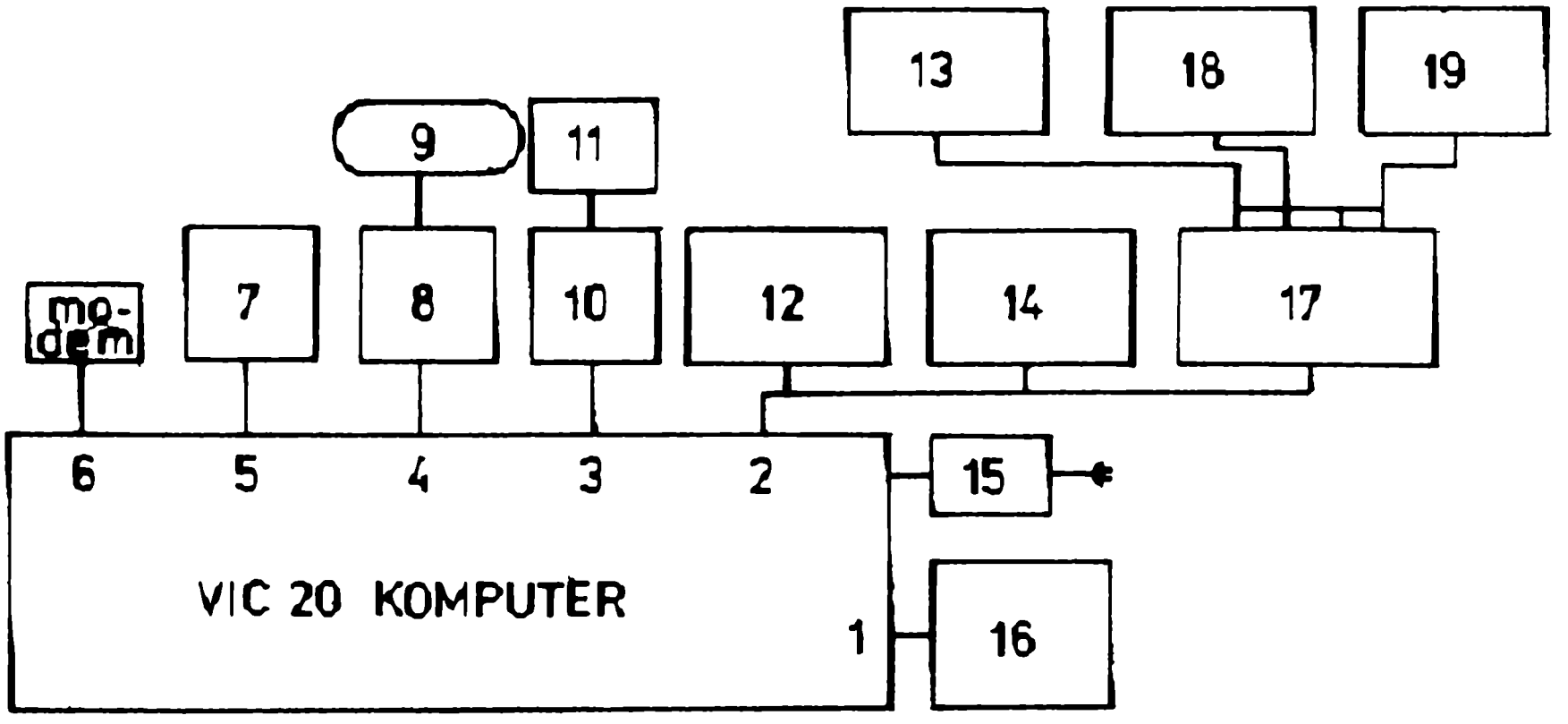
A) MELLÉKLET: A VIC TARTOZÉKAI

RÖVID ISMERTETÉS

Mivel ezt a kézikönyvet kezdőknek szántuk, nem áll szándékunkban hosszabb időt szentelni a perifériáknak, melyek a készülék hátoldalára csatlakoztathatók. Minden perifériális egységnek van saját gépkönyve, mely leírja, hogyan kell a VIC-hez csatlakoztatni.

A VIC-et úgy tervezték, hogy ki tudja elégíteni az ön igényeit. Van egy általános séma a VIC és a különböző periférиák csatlakoztatására, melyet a következő ábra mutat.

COMMODORE VIC 20 SZEMÉLYI SZÁMÍTÓGÉP RENDSZER



1. Játék bemenet; 2. Bővítő bemenet; 3. Videó kimenet; 4. Soros busz; 5. Kazetta bemenet; 6. Felhasználói bemenet; 7. Kazettás egység; 8. Egyes floppy meghajtó; 9. Nyomtató; 10. RF vagy Videó kábel; 11. TV vagy monitor; 12. S3 memória bővítő; 13. 3K/8K/16K memória bővítő; 14. Dugaszolható programok; 15. Tápegység; 16. 1 Josystick vagy 1 fényceruza vagy 2 paddle; 17. "Master" vezérlő kártya; 18. Dugaszolható programok; 19. Interface cartridge

SPECIÁLIS DUGASZOLHATÓ PROGRAMOK

Szuper bővítő:

- 3K bővítő memória (a VIC-et 8K-ra bővíti)
- Nagy felbontású grafika és rajzutasítások
- Előre meghatározott függvény billentyűk

Programozást támogató:

- Programozási eszköz: "Kit"
- Gépi kódú monitor
- Előre meghatározott függvény billentyűk (program, utasítások)
- Felhasználó által meghatározott függvény billentyűk.

SZALAGOS KAZETTA EGYSÉG

Valószínűleg a Commodore magnetofon az első periféria, amelyet beszerez. A magnetofon több ezer karaktert (betűk, számok) képes tárolni egy közönséges kazettán. Programokat tárol a szalagon, és vissza tudja őket tölteni anélkül, hogy azokat minden alkalommal be kellene gépelni. (ld. B) Melléklet: A kazettás magnó kezelése)

A VIC kazettás egységet csatlakoztassa a saját csatlakozójával a VIC hátoldalán található Casette Interface aljzatba — nem szükséges egyéb külön interface.

Számos előre elkészített játékprogram vásárolható meg kazettán.

MASTER VEZÉRLŐ KÁRTYA

A VIC hátoldalán található bővítő bemenetre csatlakoztatható a Master vezérlő Kártya. Ez a kártya lehetővé teszi, hogy egyszerre több cartridge-ot használjon. Ennek további hat csatlakozója van, mely képes program cartridge-okat, memória bővítőket fogadni, tartalmaz egy IEEE 488 interface-t, mely lehetővé teszi a Commodore PET/CBM perifériák és más IEEE eszközök használatát.

VIC FLOPPY LEMEZ EGYSÉG

A VIC floppy lemez 170 000 karakterig (számok és betűk) és nagyon hosszú programokat képes ki és beírni a VIC memóriájába a másodperc tört része alatt. Nincs hosszú várakozás a program keresésekor, olvasásakor, betöltésekor. Ez az egység a VIC soros bemenetére csatlakozik.

IEEE 488 INTERFACE CARTRIDGE

Az IEEE 488 egy univerzális tudományos szabvány, mely lehetővé teszi a Commodore PET/CBM perifériák, mint a diszk meghajtó, nyomtató, valamint egyéb tudományos eszközök használatát.

SOROS NYOMTATÓ

A VIC rendelkezik egy soros interface-szel, mely lehetővé teszi a számítógéppel a soros vonalon történő kommunikálást. A Commodore speciális pont mátrix nyomtatója ezt a soros vonalat használja, s így ki tudja nyomtatni a programot és az eredményeket.

JÁTÉK BEMENET

A VIC játék bemenete lehetővé teszi joystick, fényceruza, potméter csatlakoztatását, és így a billentyűzet használata nélkül könnyen lehet izgalmas játékot játszani.

TELEKOMMUNIKÁCIÓ

Biztosan hallott olyan programokról, amelyek segítségével a személyi számítógép más számítógépekhez fordul információkat szerezni a raktárkészletről, üzleti életről vagy egyéb dolgokról. Ezt a lehetőséget a VIC-be is beépítették. A VIC felhasználói bemenete az R5232 interface-t tartalmazza, mely lehetővé teszi az információ cserét telefonvonalon keresztül, egy olcsó telekommunikációs modem segítségével.

SOTWARE

Ezeken a hardware-eken túlmenően a VIC rengeteg érdekes software-rel rendelkezik.

Néhány software programot hardware cartridge-on tárolnak, melyek a VIC hátoldalához csatlakoztathatók, és bekapcsolás után azonnal futtathatók. Néhány szalagon, diszken található, és nagyon sok van a VIC tanuló sorozat könyveiben, melyek a programozás elsajátításában nagy segítséget nyújtanak.

Ez csak rövid áttekintés volt a kapható tartozékokról.

B) MELLÉKLET: A KAZETTÁS MAGNÓ HASZNÁLATA

A kazettás magnó az első periféria, amelyet a legtöbb VIC tulajdonos megvásárol. Ez az egység lehetővé teszi, hogy felvegyen programokat későbbi felhasználásra, tároljon, visszanyerjen adatokat. Ez a fejezet segít önnek, mihelyt megvan a kazettás magnója.

Az első utasítás, amelyet használni fog a magnóval, a SAVE utasítás. Amennyiben ezt az utasítást adta a VIC-nek, a VIC a következő üzenettel válaszol: PRESS PLAY § RECORD ON TAPE (nyomja meg a felvétel és lejátszó gombot). Tehát most le kell nyomnia egyszerre a felvétel (RECORD) és lejátszó (PLAY) gombot. Mindkét billentyűnek lenyomva kell lenni, mialatt a VIC felveszi az ön programját. (Amennyiben a RECORD gombot nem lehet lenyomni, ellenőrizze, hogy a kazetta nincs-e hátul kitörve. Ebben az esetben a VIC nem vesz fel a magnetofonra).

Ha a PLAY gombot lenyomta, a VIC kiír egy OK üzenetet, majd a SAVING (mentés) szót írja ki a képernyőre. Rövid időn belül megjelenik a OK üzenet és a READY (kész) szó a képernyőn, ezzel a SAVE folyamat véget ért. A programnak nevet adhat, melyet idézőjelbe tesz a SAVE után. (ld. C) Melléklet)

Miután a programot elmentette, célszerű ellenőrizni, hogy a szalagra felvett program nem hibás-e. Előfordulhat például, hogy a szalag hibás. Éppen ezért a VIC-nek van egy másik utasí-

tása, a VERIFY (felülvizsgálat). Ez az utasítás összehasonlítja a szalagon levő programot a RAM memóriában találhatóval.

Miután begépelte a VERIFY szót, a VIC kiírja a képernyőre: PRESS PLAY ON TAPE (Nyomd meg a Play gombot a magnón). Amennyiben a VERIFY szót a program neve követi, a magnó megkeresi azt a programot (különben az első programot ellenőrzi a szalagon) Miután lenyomta a PLAY gombot, a VIC nemcsak egy OK-val és a SERACHING (keresés) szóval válaszol. A VIC nemcsak azokat a programokat listázza ki nevűkkel, amelyeket keresi, hanem a többit is, hogy tudja, azok is ott vannak. Miután megtalálta a keresett programot, kiírja a FOUND (megtalálva) és a VERIFYING (ellenőrzés) üzenetet. Ekkor a VIC a szalagon levő programot összehasonlítja a memóriában levővel. A programot a végéig ellenőrzi, függetlenül attól, hogy korábban már hibát talált. Miután eljutott a program végéig az OK és READY üzenetet írja ki, ha minden rendben van. A VERIFY ERROR (ellenőrzési hiba) jelenik meg a képernyőn, ha hibát talált a kazettán levő programon. Amennyiben ilyen hibajelzés van, próbálja meg újra a SAVE VERIFY műveletet egy másik kazettával.

Amennyiben kazettáról kívánunk programot betölteni a VIC belső memóriájába, a LOAD parancsot használjuk. A VIC közli önnel, hogy nyomja meg a lejátszó gombot a magnón (PRESS PLAY ON THE TAPE), és a program betöltődik a VIC-be. Amikor ez helyesen megtörtént, READY üzenetet ír ki a képernyőre, ha viszont hiba van a szalagon levő programnál, a LOAD ERROR (betöltési hiba) jelenik meg. Ha a szalag nagyon hibás, furcsa dolgok történhetnek a VIC-kel. Például mindenféle karakter jelenhet meg a képernyőn. Ha ilyen jellegű hibás betöltés történik, a legcélszerűbb kikapcsolni a készüléket, majd újra bekapcsolni és újra megpróbálni az egészet.

Most pedig eljutottunk a kazettás egység egy sokkal bonyolultabb használatához. Ez pedig az adatok tárolása szalagon és onnen történő visszanyerésük.

Az első lépés, hogy adatokat tárolhassunk a szalagon, egy file nyitása. Ez az OPEN utasítás, melyet három szám és egy név követ, a következőképp:

```
OPEN 1, 1, 1, "HELLO"
```

Az OPEN szót követő első szám a file szám, melyre később a programban utalunk. A 2. szám a fizikai egység száma, amelyik 1 a kazettás magnó esetén, míg a harmadik 0 olvasás, 1 írás, 2 írás szalagvég felírással együtt. Vegyük észre, hogy az íráskor nem keres üres helyet, és olvasáskor a VIC az adott nevű file-t keresi meg. A fentebb bemutatott OPEN utasítás egy file-t nyit meg a szalagon az írásokhoz, melyet hello-nak nevez el a szalagon, és melyre a program későbbi utasításai folyamán # 1-el fog hivatkozni.

Az adatok pillanatnyi felírására a PRINT # utasítás vonatkozik. A # jel, melyet a file szám követ (a fenti példában 1), majd egy vessző és a szalagra írandó adatok listája. A lista ugyanolyan formában megy ki a képernyőre, mint amikor PRINT-et használunk. A vessző extra szüneteket eredményez az egyes értékek között, így felesleges szüneteket generál a szalagon, mivel az üres karakter épp annyi helyet foglal, mint a többi. Általában a pontosvesszőt használják ebben a sorban elválasztó jelként.

Észre fogja venni, hogy file íráskor a szalagra, a kazetta nem folyamatosan fut, hanem elindul, majd megáll újra elindul és így tovább. Ez azért van, mert a VIC puffer memóriában tárolja az adatokat a kiírásnál. Ez azt jelenti, hogy az adatokat nem követlenül írja ki a szalagra, ha nem leteszi a RAM egy speciális területére, melyet puffernak hívunk. A puffer 192 karaktert

karaktert tud tárolni. Mikor a 192. karaktert is betettük a pufferba, a VIC megállítja a további karakterek bevitelét a pufferba, és a puffer teljes tartalmát mint egy blokkot kiviszi a szalagra.

Miután befejezte az adatok kiírását a szalagra, le kell zárni a file-t a CLOSE utasítással. A CLOSE szót az a file szám követi, melyet le kíván zárni. Ez a puffer miatt fontos. Ha vannak még karakterek a pufferban, és befejezte az írást, enélkül ezek a karakterek elvesznének. A CLOSE hatására az összes pufferban maradt karakter kiíródik a szalagra, valamint felviszi a file-vég (szalagvég, amennyiben ilyen módban van) jelet is a szalagra.

Ahhoz, hogy az adatokat visszanyerjük a szalagról, először meg kell nyitni a file-t a fentiek szerint, és az OPEN-t követő harmadik számnak 0-nak kell lenni. A VIC megkeresi a file nevet, amennyiben korábban megadtuk, és a szalag megáll, amikor megtalálta. Most az INPUT # vagy a GET # utasítást használjuk az adatok bevitelére. Amennyiben az INPUT # vagy GET # utasításnál a puffer üres, egy puffernyi adat lesz betöltve a szalagról. Az adatokat a pufferból olvassa ki, nem pedig közvetlenül a szalagról.

Az INPUT # és GET # utasítás között az a különbség, hogy az INPUT # olvas egy teljes változót, legyen az szám vagy string a szalagról, míg a GET # csak egy karaktert vesz be egyszerre.

Természetesen, ha megfigyeli a GET # utasítást, észreveszi, hogy minden adat utolsó karaktere a CHR$ (13), amely nem más, mint a kocsi vissza. Ezt a karaktert használják az adatok elkülönítésére a szalagon, és ily módon tudja az INPUT utasítás, hogy hol van vége a változónak.

Bemutatunk 3 programot, mellyel gyakorolni lehet a file írását és olvasását. Az első program egy file-t nyit meg. A második programot azután írjuk be, miután az első már fut, és az adatszalagot visszatekertük. Ez a program az INPUT használatát mutatja be, az információ gyűjtéskor. A 3. program a GET utasítást mutatja be, hogyan lehet adatot beolvasni karakterenként, és nemcsak az adatot, hanem annak az ASCII kódját is kiírja.

PROGRAM 1: szalagra írás

```
10 PRINT " [SHIFT] [CLR HOME]  SZALAGRA IRÓ PROGRAM"
20 OPEN 1, 1, 1, "ADAT FILE"
30 PRINT "KÉREM AZ ADATOT, VAGY"
40 PRINT "STOP, HA VÉGE"
50 PRINT
60 INPUT "ADAT"; A$
70 PRINT # 1, A$
80 IF A$ <> "STOP" THEN 50
90 PRINT
100 PRINT "FILE LEZÁRÁS"
110 CLOSE 1
```

PROGRAM 2: szalagról olvasás INPUT # segítségével

```
10   PRINT " [SHIFT] [CLR HOME]    OLVASÁS SZALAGRÓL"
20   OPEN 1, 1, 0, "ADAT FILE"
30   PRINT "FILE MEGNYITÁS"
40   PRINT
50   INPUT # 1, A$
60   PRINT A$
70   IF A$ = "STOP" THEN END
80   GOTO 50
```

PROGRAM 3: szalagról olvasás GET # segítségével

10–40. ugyanaz, mint PROGRAM 2

```
50   GET # 1, A$
60   IF A$ = "" THEN END
70   PRINT A$;
80   GOTO 50
```

C) MELLÉKLET: VIC BASIC

Ez a kézikönyv eddig a BASIC programozási nyelvbe adott betekintést – elegendően ahhoz, hogy kellő érzéket szerezzünk a számítógép programozáshoz, és megismerjük az ehhez használható „szókészletet" Ebben a mellékletben teljes listát találunk a VIC BASIC valamennyi szintaktikai szabályáról, rövid leírással. Próbáljuk csak ki nyugodtan az utasításokat. Nem okozhatunk tartós kárt a számítógépbe, ha programokat adunk be; programozni pedig csak úgy lehet a legkönnyebben megtanulni, ha magunk csináljuk.

A BASIC műveletek különböző fajtáinak megfelelően ez a melléklet több részre tagozódik:

1. *Változók és műveletek:* az a rész a különböző változótípusokat, érvényes változóneveket, aritmetikai és logikai műveleteket írja le.
2. *Parancsok:* az a rész a parancsokat írja le, amelyek a programok szerkesztéséhez, töltéséhez, mentéséhez és törléséhez használatosak.
3. *Utasítások:* ez a rész azokat a BASIC utasításokat írja le, amelyek számozott programsorokban előfordulnak.
4. *Függvények:* ez a rész a string-, numerikus és nyomtatási függvényeket írja le.

Ebben a fejezetben az utasításokat ABC sorrendben vesszük végig.

1. VÁLTOZÓK ÉS MŰVELETEK

a) VÁLTOZÓK

A VIC 3 típusú változót használ a BASIC-ben. Ezek a következők: normál numerikus, egész numerikus, string (alfanumerikus) változók.

Normál numerikus változók, más néven lebegő pontos változók -10^{38} és $+10^{38}$ között vehetnek fel értéket 9 számjegy pontossággal. Amennyiben a szám nagyobb lesz, mint amit 9 számjegy ábrázolni tud, azaz 10^{10} és 10^{-10}-en belül, a computer átvált exponenciális kijelzésre, azaz a számot normalizálja 1 egész és 8 tizedes helyre, melyet az E betű követ, és 10 hatványai, mellyel a számot szorzzuk. Például az 12345678901-et a következőképp jelzi ki: 1.23456789E + 11

Egész típusú változók, ezek értéke + 32767 és −32768 között lehet. Az egész típusú változók kevesebb memóriahelyet igényelnek, mint a lebegő pontosak, de az eltérés nem olyan számottevő, kivéve, ha nagyobb méretű tömbváltozókat használunk. Az egész típusú változók a következők:

5 , 10 , − 100

String változók, ezeket karakstereknél használjuk, mely tartalmazhat számokat, betűket és egyéb karaktereket, melyeket a VIC előállít. Példa string változóra: "VIC 20"

A változó neve lehet egy betű, két betű vagy betű és azt követően egy szám.

Az egész típusú változót a változó neve utáni százalékjel (%) jelzi. A string változót pedig a neve után írt dollárjel ($).

Példák:

Numerikus változó nevek: A, A5, BZ

Egész típusú változó nevek: A%, A5%, BZ%

String változó nevek: A$, A5$, BZ$

Tömbváltozók: a változók egy olyan sorozata, melyeknek ugyanaz a nevük, plusz még egy szám vagy több, mely meghatározza, melyik melyik. Ezeket a DIM utasítással definiáljuk. Tartalmazhatnak lebegő pontos, egész típusú vagy string változókat. A tömb változók nevét egy (vagy több) zárójelben levő szám követi, mely a változó helyét határozza meg a tömbön belül.
Példák: A(7), BZ%(11), A$(87)

A tömbök több dimenziósak is lehetnek. A két dimenziós tömbök sorokból és oszlopokból állnak, ahol a zárójelen belül álló első szám határozza meg a sor számát, a második pedig az oszlopszámot.

Példák: A(7,2), BV%(2, 3, 4), Z$/3, 2

Megjegyzés: Létezik három olyan változónév, amelyet a VIC maga használ, és így ezeket nem definiálhatjuk: ST, TI és TI$.

ST a státusváltozó értékét az előzőleg elvégzett input, vagy output műveletnek megfelelően kapja. Az ST értéke megváltozik például akkor, ha mágnesszalagnál és floppynál az íráskor vagy olvasáskor probléma merül fel.

TI értékét a belső óra minden 1/60 másodpercben eggyel megnöveli. Ez az érték a számító gép bekapcsolásakor Ø, és csak akkor állítható vissza, ha a TI$ megváltozik.

A TI$ egy hat számjegyből álló string, amelyet a rendszer a mindenkor érvényes értékre állít. Az első két számjegy az órákat jelenti, a két középső a perceket, az utolsó kettő pedig a másodperceket. Ennek a változónak tetszőleges értéket adunk, amely attól kezdve folyamatosan nőni fog.

TI$ = "101530" az órát 10 óra 15 perc 30 másodpercre állítja.

Az óra a számítógép kikapcsolásával törlődik, és bekapcsolásakor Ø-val indul.

b) MŰVELETI JELEK

Az aritmetikai műveleteket a következő műveleti jelekkel képezzük:

\+ összeadás

— kivonás

* szorzás

/ osztás

↑ hatványozás

Ha egy aritmetikai kifejezésben több aritmetikai műveleti jelet használunk, akkor először a hatványozás hajtódik végre, majd a szorzás az összeadás és kivonás előtt.

A számítás sorrendje megváltoztatható zárójelek segítségével, mivel a zárójelbe tett műveletek az összes többi előtt hajtódnak végre. Ellenőrizze a képletnél, hogy a nyitó zárójelek száma megegyezik a záró zárójelekével, mert különben a program futás közben SYNTX ERROR üzenetet ír ki.

A feltételvizsgálatokban a következő jelek használhatók:

= egyenlő

< kisebb, mint

> nagyobb, mint

<= kisebb, vagy egyenlő

>= nagyobb, vagy egyenlő

<> nem egyenlő

Végezetül vannak még logikai műveletek:

AND		ÉS (két feltétel egyidejűleg teljesül)
OR		VAGY (két feltétel közül legalább az egyik teljesül)
NOT	=!	NEM (a feltétel nem teljesül)

Ezeket általában olyankor használjuk, ha egy IF ... THEN utasításban többszörös feltételt akarunk elhelyezni.

Például:

IF A = B AND C = D THEN 100 (mindkét feltételnek teljesülnie kell),

IF A = B OR C = D THEN 100 (a két feltétel közül legalább az egyiknek teljesülnie kell).

2. PARANCSOK

CONT (continue)

Ezt a parancsot használjuk az olyan program folytatására, amelynek futása a STOP gomb benyomásának következtében, vagy a programon belül egy STOP utasítás, illetve END utasítás hatására szakadt meg. A program végrehajtása ugyanazon a helyen folytatódik, ahol a megszakítás történt.

A CONT parancs nem működik, ha előtte megváltoztattunk vagy beszúrtunk egy programsort. Akkor sem működik a CONT parancs, ha a programmegszakítás hiba miatt következett be, vagy ha a CONT előtt valamilyen hibás parancsot próbáltunk végrehajtani. Ezekben az esetekben a CAN'T CONTINUE ERROR hibajelzést kapjuk.

LIST

A LIST utasítás lehetővé teszi, hogy sorokat nézzen meg a BASIC programban, melyet begépelt vagy betöltött a VIC memóriájába. Amennyiben egyedül áll, nem követi semmilyen szám, úgy a program teljes litája megjelenik a képernyőn (melynek futása lassítható a CTRL billentyűvel, vagy megállítható a STOP gomb lenyomásával, amelyet a RUN/STOP jelöl). Amennyiben a LIST szót egy sorszám követi, akkor a VIC csak azt a sort mutatja meg. Ha a LIST-et 2 szám követi kötőjellel elválasztva, úgy a VIC a 2 sorszám közötti összes sort kiírja. Amennyiben a LIST-et egy szám és utána egy kötőjel követi, úgy ettől a sorszámtól a program végéig fog listázni. Ha a LIST-et egy kötőjel, majd egy szám követi, úgy a listázás a program elejétől eddig a sorszámig tart. Próbálja ki ezeket a variációkat, így ellenőrizni tud bármely programrészt, vagy kiírhat egy sort változtatás céljából.

Példa:

LIST	a teljes programot kiadja
LIST 10–	kiadja a programot a 10-es sortól végig
LIST 10	csak a 10-es sort adja ki
LIST –10	kiadja a programot az elejétől a 10-es sorig
LIST 10 – 20	a 10-es sortól kezdődően kiadja a program adott részletét a 20-as sorig bezáróan.

LOAD

Ezt az utasítást használjuk egy programnak szalagról vagy floppyról a számítógépbe való betöltésére. Ha csak LOAD-ot adunk be, és utána megnyomjuk a RETURN gombot, a következő megtalált programot fogja a számítógép a kazettáról beolvasni. A parancs kiegészíthető egy idézőjelek közé tett programnévvel. A nevet követheti egy vessző és egy szám vagy numerikus változó, amely a készülékszámot jelenti, és azt adja meg, hogy melyik készülékről töltendő a program.

Ha nem adunk meg készülékszámot, a VIC automatikusan az 1-es számú készülékhez, a kazettás magnóhoz fog fordulni. A másik, LOAD paranccsal megadható készülék a floppy-egység, a 8-as számú készülék.

LOAD	— a következő programot olvassa a kazettáról
LOAD "HALLO"	— megkeresi a kazettán a HALLO programot, és ha megtalálta, betölti
LOAD A$	— megkeresi azt a programot, amelynek neve az A$ változó alatt van tárolva
LOAD "HALLO", 8	— a HALLO nevű programot keresi meg a floppyn
LOAD " * ", 8	— megkeresi az első programot a floppyn

A LOAD parancsot használhatjuk a BASIC programon belül is, a következő szalagon található program betöltésére és futtatására.

NEW

Ez a parancs a számítógép központi tárolójában levő programot törli. Ugyancsak törlődik ekkor valamennyi esetleg használt változó is. Ha a programunkat előtte nem mentettük el SAVE segítségével, akkor elveszett. Ezért körültekintően használjuk ezt az utasítást!

A NEW-t programon belül is használhatjuk, ha azt akarjuk, hogy a program végrehajtása után a számítógép ismét szabad legyen új programok számára. Amikor elérjük ezt az utasítást, a programot törli, és minden megáll.

RUN

Ez a parancs elindítja a program végrehajtását, ha az be lett töltve a számítógép központi tárolójába. A címke nélkül megadott RUN a legkisebb címkével ellátott sorral kezdi a végrehajtást. Ha a RUN után egy címkét is megadunk, a program az így megadott soron kezd el futni.

RUN	elindítja a programot a legkisebb címkén
RUN 100	elindítja a programot a 100-as soron
RUN X	SYNTAX ERROR. A címkének létezőnek kell lennie, és explicit alakban kell megadni. A változók nem megengedettek.

SAVE

Ez a parancs a számítógépben levő programot szalagra vagy floppyra menti. Ha csak SAVE-et adunk be, és megnyomjuk a RETURN gombot, a program a mágnesszalagra mentődik. A számítógép nem tudja megállapítani, hogy a szalagnak ezen a részén van-e már program. Éppen ezért körültekintően járjunk el, különben tönkretehetünk egy értékes programot.

Ha a SAVE után idézőjelek között egy nevet is megadunk, vagy egy string változót, akkor a program ezt a nevet fogja kapni, és így a jövőben könnyebb lesz megkereshető és azonosítható. A nevet egy vessző követheti (az idézőjel után) és egy szám numerikus változó. Ez a szám megmondja, milyen készülékre lesz eltárolva a program. Amennyiben a készülékszám 1, úgy a kazettás egységről, ha 8, akkor a diszk egységről van szó.

A készülékszám után, vesszővel elválasztva még egy második szám is következhet, mégpedig 0 vagy 1. Ha ez a szám 1, akkor a VIC a szalagra egy END - OF - TAPE jelet fog írni a program után. Ez jelzi leolvasáskor a számítógépnek, hogy a LOAD után csak eddig a helyig kell keresni. Ha LOAD utasítást adunk, és a számítógép megtalálja ezt a jelet, még mielőtt a kívánt programot megtalálta volna, FILE NOT FOUND ERROR hibajelzést ad.

SAVE	név nélkül szalagra menti a programot
SAVE "HALLO"	HALLO név alatt elmenti a programot szalagra
SAVE A$	a szalagra menti a programot az alatt a név alatt, amely az A$-ben van rögzítve.
SAVE "HALLO", 8	HALLO név alatt floppyra menti a programot
SAVE "HALLO", 1, 1	kazettára menti a programot HALLO név alatt, és elhelyez utána egy szalagvég jelet

VERIFY

A VERIFY utasítja a számítógépet, hogy a központi tárolóban levő programot összehasonlítsa a szalagon vagy floppy-n levő programmal. Ezzel győződhetünk meg arról, hogy a program a SAVE-vel korrekt módon el lett mentve, és ennek során a szalagon, illetve lemezen nem keletkezett hiba. Ez az utasítás a szalag pozicionálásánál is nagyon hasznos, ekkor a VIC a szalagon található utolsó program után ír. A tennivaló a következő: közli a VIC-kel, hogy ellenőrizze (VERIFY) a szalagon található utolsó programot. Ezt a gép megteszi, majd közli, hogy a két program nem egyforma (amit már tudtunk). Most a szalag ott áll, ahol kívántuk, és fel tudja venni a következő programot az előző törlésének a veszélye nélkül. Pusztán a VERIFY megadásával a VIC a szalagon megtalált első programot fogja összehasonlítani a központi tárolóban levő programmal, névtől függetlenül.

Ha a VERIFY-t programnév vagy string változó követi, a gép megkeresi az ilyen nevű programot, és ellenőrzi. Megadhatunk továbbá készülékszámot is, melyen ellenőrzi a programot (1 magnóra, 8 diszkre).

Példa

VERIFY	ellenőrzi a következő programot a kazettán
VERIFY "HALLO"	megkeresi a HALLO nevű programot a kazettán, és összehasonlítja a központi tárolóban levő programmal

VERIFY "HALLO", 8 lemezen keresi meg és ellenőrzi a HALLO nevű programot.

3. UTASÍTÁSOK

CLOSE

Ez az utasítás befejez és lezár egy file-t, amely OPEN-nel meg lett nyitva. A CLOSE után álló szám annak a file-nak a száma, amelyet le kell zárni.

Példa:

CLOSE 2 csak a 2-es file-t zárja le.

CLR

Ez az utasítás a tárolóban az összes változót törli, magát a programot azonban érintetlenül hagyja. A CLR automatikusan végrehajtódik akkor is, ha RUN-t adtunk.

CMD

A CMD azokat a kiírásokat amelyek normális körülmények között a képernyőre mennének (pl. PRINT vagy LIST, POKE azonban nem), helyette egy másik készülékre küldi. Ez lehet a nyomtató vagy egy adatfile a kazettán vagy floppyn. A készüléket vagy file-t előzetesen meg kell nyitni (OPEN). A CMD-t követnie kell egy számnak vagy numerikus változónak, amely a file-számra vonatkozik.

Példa:

OPEN 1, 4	megnyitja a 4-es készüléket, a nyomtatót
CMD 1	minden normál kiadás a nyomtatóra menjen
LIST	a program listáját adja ki, de most már nem a képernyőre, hanem a nyomtatóra.

Ha a további kiírásokat már ismét a képernyőre kívánjuk, töröljük a CMD 1 hatását a file bezárásával.

DATA

Ezt az utasítást olyan adatok listája követi, melyeket a READ használ. Az adatok lehetnek numerikusak vagy stringek. Egymástól vessző választja el őket. A stringeket nem kell idézőjelbe tenni, csak akkor, ha space-t, kettőspontot vagy vesszőt tartalmaznak. Ha két vessző között semmi sincs, akkor az olvasott érték Ø, illetve üres string lesz.

Példa:

10 DATA 100, 200, PISTA "JÓ NAPOT", 3.14, ABC123

Mivel a programban szinte soha nincs szükség a DATA utasítás azonnali végrehajtására információszerzés céljából, célszerű a DATA utasítást (amennyire csak lehet) a program végére tenni. Ez gyorsítja a program futását.

DEF FN

Bonyolult számításokat definiálhatunk függvényként egy rövid névvel. Olyan bonyolult képlet esetén, amelyet sokszor használunk, sok helyet és időt tudunk így megtakarítani.

A függvény neve FN, kiegészítve bármely megengedett változónévvel (1 vagy 2 karakter). A függvényt a DEF utasítás definiálja, amelyet a függvény neve követ. A név után egy numerikus változót kell zárójelben megadni (jelen esetben X). Azután következik egy egyenlőségjel, majd az aktuális képlet, amelyben a változó szerepel, a megfelelő helyen. A függvény most már meghívható, amennyiben a változónak valamilyen értéket adtunk, vagy más változóval helyettesítettünk.

```
10 DEF FNA (X) = 12 * (34.75-X) .3
20 PRINT FNA (7)
```

A példa eredménye 137.

Megjegyzések: a csillag a szorzást szimbolizálja. A 7-es számot a képletben található X helyére írjuk.

DIM (egy tömb dimenzionálása)

Mielőtt elkezdené a tömbváltozókat használni, és azok elemszáma nagyobb, mint 11, úgy a programon belül egy DIM utasítást kell végrehajtani az adott tömbre vonatkozólag. A DIM utasítást a tömb neve követi, mely bármely engedélyezett változónév lehet. Ezekután zárójelben a tömb elemszáma (lehet változó is) következik dimenzióként. Az egy dimenziósnál nagyobb tömböt mátrixnak nevezzük. Bármilyen dimenziószám használható, de ne felejtse el, hogy a változók teljes sorozata egy nagy területet foglal el, és így könnyen kifuthat a memóriából. A dimenzionálás által meghatározott elemek számát a tömb különböző dimenzióiban találhatók szorzata adja meg.

Példa

```
10    DIM A$ (40), B7 (15), CC% (4, 4, 4)
```

41 elem 16 elem 125 elem

Egy DIM utasításon belül több tömböt is lehet dimenzionálni. Ezeket a tömböket vesszővel választjuk el egymástól. Ügyeljünk arra, hogy a program egy tömböt csak egyszer dimenzionáljon, mert különben hibajelzést kapunk. Legcélszerűbb a DIM utasításokat a program elejére tenni.

END

Ha a programban a gép egy END-et talál, megállítja a programot, mintha az utolsó sor végére ért volna. Ilyenkor a CONT beadásával indítható újra.

FOR ... TO ... STOP

A NEXT utasítással együtt ez az utasítás arról gondoskodik, hogy egy programrészlet többször végrehajtódjék.

Ha azt akarja, hogy a VIC elszámoljon sokáig, vagy ha valamit megszámoltatni kíván, akkor a program várakozzék néhány másodpercig. Ez a BASIC egyik legáltalánosabban használt utasítástípusa.

Az utasítás formátuma a következő:

FOR (a ciklusváltozó neve) = (kezdőérték) TO (végérték).

A ciklusváltozó az a változó, melyet hozzáadunk vagy levonunk a kezdő értékből a program során. A kezdő és végérték a ciklusváltozó értékeinek a határa.

A FOR utasítás logikája a következő. Először beállítja a ciklusváltozó értékét a kezdő értékre. A végértéket a VIC eltárolja egy későbbi összehasonlítás végett. Miután a program eléri a NEXT utasítást tartalmazó sort, a ciklus változó értékéhez egyet hozzáad, és ellenőrzi, hogy az érték nem nagyobb-e, mint a végérték. Ha nem nagyobb, akkor a FOR utasítás után következő sort hatja végre. Amennyiben a ciklusváltozó értéke nagyobb, mint a végérték, úgy a NEXT utasítás után következő sort hajtja végre.

Példa

```
10    FOR L = 1 TO 10
20    PRINT L
30    NEXT L
40    PRINT "KÉSZ VAGYOK! L = "L
```

A program kiírja a számokat 1-től 10-ig a képernyőre, majd a KÉSZ VAGYOK! L = 11 üzenet következik.

Érti, hogyan működik? Ha még nem, olvassa el a leírást a program előtt még egyszer, és kövesse végig papíron lépésenként.

A ciklus végértékét követheti a STEP szó és egy szám vagy változó. Ebben az esetben a ciklus változóhoz hozzáadódik a lépésnagyságként megadott érték. Ez lehetővé teszi, hogy visszafelé számoljunk, ne csak egész értékkel, ahogy ezt a program megkívánja.

Cikluson belül is képezhetünk ciklusokat. Ezt úgy hívjuk, hogy egymásba skatulyázott ciklusok. Ügyelni kell az egymásba skatulyázott ciklusoknál arra, hogy annak a ciklusnak kell korábban befejeződni, amelyik később kezdődött.

Példa egymásba skatulyázott ciklusra:

```
10    FOR L = 1 TO 100
20    FOR A = 5 TO 11 STEP 2
30    NEXT A
40    NEXT L
```

Megjegyzés: a 20., 30. sorszámú ciklus van beágyazva a nagyobba.

Hibás:

```
10   FOR L = 1 TO 100
20   FOR A = 5 TO 11 STEP 2
30   NEXT L
40   NEXT A
```

GET

A GET utasítás segítségével adatok vehetők át a billentyűzetről, mégpedig alkalmanként egy karakter. A GET végrehajtása során a beadott karakter hozzárendelődik a változóhoz. Ha nem adtunk be semmit, üres jel vagy nulla rendelődik hozzá.

A GET után egy változónév áll, amely általában string változóhoz. Ha numerikus változót használunk, és egy nem numerikus gombot nyomunk le, hibajelzéssel megszakad a program.

A GET utasítást olyan ciklusba célszerű elhelyezni, amely tartalmazza az "üres" eredmény vizsgálatát. Az alábbi ciklus mindaddig várakozik, amíg valamelyik billentyűt le nem nyomjuk.

Példa:

```
10 GET A$ : IFA$ = " "THEN 10
```

Megjegyzés: Ez a sor arra vár, hogy egy billentyűt leüssünk. Bármilyen gomb megnyomására folytatódik a program.

GET

A GET # arra használatos, hogy egy előzőleg OPEN-nel megnyitott file-ból vagy készülékről egy jelet beolvassunk.

GET #1, A$ beolvas egy jelet egy adatfile-ból.

GOSUB

Ez az utasítás hasonlóan működik, mint a GOTO, a számítógép azonban megjegyzi, hogy a program melyik helyén kapta a GOSUB utasítást. Ha a program elérkezik egy RETURN utasításhoz, visszaugrik a közvetlenül a GOSUB utáni helyre. Ez különösen akkor hasznos, ha egy programrészt gyakrabban használunk. Ezeket a programsorokat ilyenkor nem kell többször beadni, hanem szükség szerint a GOSUB-bal "meghívhatók".

```
20 GOSUB 800
```

Ez azt jelenti, hogy ugorjon a 800. sornál kezdődő szubrutinra, és hajtsa végre.

GOTO vagy GO TO

A program a GOTO után megadott címkére ugrik, és ott folytatja a munkát. Ha a GOTO után megadott címke nem fordul elő a programban, a következmény programmegszakítás hibajelzéssel.

IF ... THEN

Az IF ... THEN egy szituáció elemzésére utasítja a számítógépet. Az elemzés eredményének függvényében dönt két lehetőség között. Ha az IF utáni feltétel teljesül (igaz), akkor a sor THEN után álló része hajtódik végre. Ez lehet egy sorszám, amely hatására a VIC arra a sorra ugrik. Ezen kívül lehet bármely tetszőleges BASIC utasítás.

Ha a feltétel nem teljesül, a program a következő sort hajtja végre. A feltétel lehet egy változó vagy egy képlet. Ez akkor számít igaznak, ha a számítás eredménye nem nulla, ellenkező esetben hamis. A legtöbb esetben feltételként olyan kifejezés áll, amely a logikai operátorok valamelyikét (=, < , > , <=, >= <> AND, OR, NOT) tartalmazza. Ha az eredmény igaz, értéke –1 és 0, ha nem.

Ennek megértéséhez lásd a logikai műveletekkel foglalkozó részt.

```
10 IF X > 0 THEN END
```

INPUT

Az INPUT utasítással a program a felhasználó által beadandó adatokat fogadja, és egy változóhoz rendeli hozzá. A program megáll, kiír a képernyőre egy kérdőjelet, és vár mindaddig, amíg a felhasználó be nem adott egy választ, és le nem nyomta a RETURN gombot.

Az INPUT után egy változónév vagy változók listája áll, ahol a változókat vessző választja el egymástól. Az INPUT és a változók között megadható idézőjelben egy üzenet a felhasználó részére. Amennyiben van üzenet (amit promt-nak neveznek), úgy a prompt utolsó idézőjele után egy pontosvessző (;) kell hogy következzék. Ha több mint egy változót kívánunk beolvasni, akkor ezeket begépeléskor vesszővel válasszuk el.

Példa:

```
10   INPUT "KÉREM GÉPELJEN BE EGY SZÁMOT"; A
20   INPUT "ÉS AZ ÖN NEVÉT"; A$
30   INPUT B$
40   PRINT "SAJNÁLOM, HOGY NEM ÉRTETTE, MIT AKARTAM"
```

INPUT

Az INPUT # hasonló az INPUT-hoz, azonban az adatokat egy előzőleg OPEN-nel megnyitott file-ból veszi.

LET

A LET az értékadást vezeti be, azonban alig használatos a programokban, mivel megadása nem kötelező. A változónév, amelyhez a számítás eredményét hozzárendeljük, az egyenlőségjeltől balra áll, a számítási képlet jobbra.

Példa:

```
10    LET A = 5
20    B = 6
30    C = A * B + 3
40    LET D$ = "HALLO"
```

NEXT

A NEXT mindig egy FOR utasítással együtt alkalmazandó. Ha a program elérkezik a NEXT utasításhoz, megvizsgálja a ciklusváltozót, hogy az előírt határt elérte-e már. Ha a ciklusnak még nincs vége, akkor a STEP után megadott érték hozzáadódik a számlálóhoz. Ha azonban elértük a határt, a program végrehajtása a NEXT után álló utasítással folytatódik.

A NEXT után állhat egy változónév vagy változók listája, amelyben a változókat vessző választja el egymástól. Először mindig a legbelső ciklusnak kell végrehajtódnia. Változók megadása esetén a kiértékelés balról jobbra halad.

Példa:

```
10    FOR L = 1 TO 10 : NEXT
20    FOR L = 1 TO 10 : NEXTL
30    FOR L = 1 TO 10 : FORM = 1 TO 10 : NEXT M, L
```

ON

Ez az utasítás a GOTO és GOSUB utasításból az IF utasítás speciális változatát állítja elő. Az ON után egy képlet következik, amelyet a számítógép ki fog számítani. A GOTO és GOSUB szavakat számok listája követi, melyeket egymástól vesszővel választjuk el. Ha a számítás eredménye 1, a listában felsorolt első címkére történik ugratás. Ha az eredmény 2, a második címkére ugrat, és így tovább. Ha az eredmény Ø, vagy nagyobb, mint a címkék felsorolt listája, akkor az ON utáni utasítás hajtódik végre.

Példa:

```
10    INPUT X
20    ON X GOTO 10, 50, 50, 50
30    PRINT "SEMMI"
40    GOTO 10
50    PRINT "HAHO"
60    ON X GOTO 10, 30, 30
```

OPEN

Az OPEN utasítás segítségével a VIC adatokat tud cserélni olyan készülékekkel, mint a kazettás magnó vagy a floppy-egység, egy nyomtató vagy akár a tv. Az OPEN kulcsszó után egy

szám (0–255), a logikai file-szám következik, amelyre minden ezt követő utasítás vonatkozik. Általában az első számot egy második követi, a készülékszám.

A készülékszámok a következők:

0	billentyűzet
1	kazetta
3	képernyő
4	nyomtató (vagy 5)
8	floppy (vagy 9–15)

Ugyancsak vesszővel elválasztva, gyakran követi a készülékszámot egy harmadik szám is, az ún. másodlagos cím. A kazetta esetében ez 0 olvasáskor, 1 íráskor, és 2 a szalagvégjeles íráskor.

A floppy esetében a másodlagos cím a puffer- vagy csatornaszámra vonatkozik. A nyomtatónál járulékos berendezéseket vezérel. A harmadik számot követheti egy string, mely lehet egy parancs a diszk meghajtónak, vagy lehet a file neve a szalagon.

Példa:

10	OPEN 1, 0	megnyitja készülékként a billentyűzetet
20	OEPN 2, 1, 0, "D"	megnyitja a magnót olvasásra, a keresett file a D
30	OPEN 3, 4	megnyitja a nyomtatót
40	OPEN 4, 8, 15	megnyitja a floppy utasításcsatornáját

Lásd még: CLOSE, CMD, GET #, INPUT #, PRINT #, ST rendszerváltozó, valamint B) melléklet.

POKE

A POKE-ot mindig két szám vagy kifejezés követi. Az első egy tárolóhely címe. 0-tól 6500-ig lehet az egyes címeket számozni. Néhányat ezek közül jól lehet használni programjainkban, mint ahogy azt a színnel és hanggal foglalkozó fejezetben már említettük. Egypárat közülük maga a VIC használ, hogy nyomon tudja követni az ön programját. A POKE utasítással való próbálkozás néhány érdekes dolgot eredményezhet. Amennyiben valami történik és ön nem tudja megállapítani, hogy mi, akkor kapcsolja ki a VIC-et majd újra be, vagy tartsa lenyomva a RUN/STOP billentyűt, és üsse le a RESTORE gombot.

A második szám 0 és 255 közötti érték, mely bekerül a memória területére, és átírja a korábban ott található értéket.

Példa:

```
10  POKE 36879,8
20  POKE 9 * 16↑13 + 15,27
```

PRINT

A PRINT az az utasítás, amelyet a kezdő elsőnek tanul meg használni. Figyelembe kell venni néhány változatát. A PRINT után állhatnak:

— szavak idézőjelek között

— változónevek

— függvénynevek

— mondatjelek

Az idézőjel közötti szavakat gyakran "másolatnak" mondjuk, mert úgy lesznek betű szerint kiírva, ahogy begépelték őket. Az idézőjelen kívüli változónevek hatására a változók pillanatnyi értéke lesz kiírva. A függvényeknek szintén a mindenkori értékük jelenik meg.

A mondatjelek az adatoknak a képernyőn való elrendezésére szolgálnak. A vessző a képernyőt két oszlopra tagolja, ezzel szemben a pontosvessző mindenféle space-t elnyom. Mindkét jel állhat utolsó szimbólumként is egy sorban. Ekkor az történik, hogy a gép a következő, PRINT-tel kiadott jelet úgy kezeli, mintha az az eredeti PRINT utasítás folytatása lenne.

```
10   PRINT "HALLO"
20   PRINT "HALLO", AS
30   PRINT A + 8;
40   PRINT J;
60   PRINT A, B, C, D
```

Lásd még a POS, TAB és SPC függvényeket.

PRINT

A PRINT és e között az utasítás között van néhány különbség. A PRINT # után egy szám következik, amely egy előzőleg OPEN-nel megnyitott készülékre vagy adatfile-ra vonatkozik. Ez után a szám után egy vessző áll, majd a kiadandók listája. A vessző és a pontosvessző ugyanazt a hatást eredményezi, mint a PRINT-nél. Ugyeljünk arra, hogy egyes készülékek nem tudnak mit kezdeni a TAB és SPC függvényekkel.
(Vigyázat: a PRINT # nem rövidíthető? # -tel!)

```
100  PRINT 1, "HALLO OTT" A$ , B$
```

READ

A READ-et arra használják, hogy a DATA sorokból adatokat rendeljünk hozzá változókhoz, hogy feldolgozhatók legyenek. Ugyelni kell arra, hogy ne string álljon ott, ahol a READ numerikus értéket vár, különben TYPE MISMATCH ERROR hibajelzést kapunk.

REM (remark, megjegyzés)

A REM után egy megjegyzés áll annak a személynek a számára, aki a programlistát olvasni fogja. A megjegyzés tartalmazhatja pl. egy programkészlet magyarázatát vagy további infor-

mációkat. A REM után bármilyen tetszőleges szöveg állhat, habár a grafikus karakterek használata furcsa dolgokat eredményezhet. A programra semmiféle hatással nincs azonkívül, hogy a hosszát megnöveli.

RESTORE

A RESTORE a DATA-sorban olvasandó következő elemre mutató pointert visszaállítja a kezdetre. Igy ezek az információk többször is olvashatók.

RETURN

Ez az utasítás zár le egy szubrutint. Ha a program egy RETURN-t talál, visszaugrik a közvetlenül a GOSUB után következő utasításra. Ha nem volt megadva a GOSUB, RETURN WITHOUT GOSUB ERROR hibajelzést kapunk. A RETURN szót semmi sem követi.

STOP

Ez az utasítás megállítja a program futását. BREAK IN xxx üzenet jelenik meg, ahol xxx az a címke, amelyen a STOP el van helyezve. A program CONT beadásával tovább futtatható. A STOP általában a debuggig, a programozási hibák keresése során használatos.

SYS

A SYS után egy szám vagy egy numerikus kifejezés következik a Ø –65535 tartományban. Ekkor a program azt a gépi kódú programot indítja el, amely ennél a tárolócímnél kezdődik. A SYS hasonló az USR-hez, nem teszi azonban lehetővé paraméterek átadását.

WAIT

A WAIT arra használatos, hogy egy programot mindaddig megállítsunk, amíg egy bizonyos tárolócím egy bizonyos értéket fel nem vesz. A WAIT szót az a szám követi, amely memória címet ellenőrizni kell. Azután egy vessző következik, majd egy másik szám. Jöhet még egy vessző és egy harmadik szám is. Az utolsó két szám értéke 0 és 255 között kell hogy legyen. Formátuma:

WAIT X, Y, Z

A megadott cím tartalmát a gép először a harmadik értékkel, Z-vel *„kizáró vagy"* kapcsolatba hozza (ha ez az utóbbi meg van adva), majd a második értékkel logikai AND-et hajt végre. Ha az eredmény Ø, a program újra megvizsgálja ennek a címnek a tartalmát. Ha az eredmény nem Ø, a program végrehajtása a következő utasítással folytatódik.

4. FÜGGVÉNYEK

a) Numerikus

ABS (X) (abszolút érték)

Az ABS egy szám abszolút értékét adja meg annak előjele nélkül (+ vagy –). A válasz mindig pozitív vagy Ø.

ATN (X) (arcus tangens)

Megadja azt a szöget (radiánban), amelynek tangense X.

COS (X) (cosinus)

Megadja az X szög cosinusát. A szöget radiánban kell megadni.

EXP (X)

Megadja az e(2.7182182) X. hatványát.

FNyy (X)

Megadja annak a felhasználó által definiált FNxx függvénynek az eredményét, amelyet egy DEFFN utasításban rögzítettünk.

INT (X)

X egész helyiértékű részét adja meg. Levágja a tizedespont utáni összes jegyet. Az eredmény mindig kisebb vagy egyenlő X-el. Ez azt jelenti, hogy a negatív számok abszolút értékűket tekintve nagyobbak lesznek (INT / –2.1) = –3.

Amennyiben az INT függvényt kerekítésre akarjuk használni, akkor a képlet a következő:

Példa:

X = INT (X * 100 + .5)) 100

Két tizedesre kerekít.

LOG (X)

Megadja X-nek az e alapú természetes logaritmusát (lásd EXP (X)). A tizedes alapú logaritmusba való átalakítás a LOG (10)-zel való osztással történik.

PEEK (X)

Megadja az X tárolócím tartalmát, ahol is X a Ø –65535 tartományba esik. Az eredmény egy egész szám a Ø –255 tartományban. A PEEK gyakran használatos a POKE-kal összefüggésben.

RND (X) (véletlen szám)

Az RND(X) Ø és 1 között egy véletlen számot generál. Ez a függvény egy véletlen (illetve álvéletlen) számot generál 0 és 1 között. Ez nagyon hasznos különböző játékoknál, lehet vele szimulálni a kockadobás vagy más elem valószínűségét, vagy egyéb statisztikai számításnál használhatjuk. Az első véletlen számot célszerű az RND (–TI) kifejezés segítségével létrehozni, hogy minden egyes programfuttatáskor új kiinduló értéket kapjunk. Később az X lehet Ø vagy egy pozitív szám.

Ha az X paraméter értéke negatív, akkor a véletlenszám-generátor számára egy új kiinduló érték képződik. Ugyanaz a negatív szám mindig ugyanazt a "véletlenszám"-sort fogja eredményezni, ha azt az RND(1) segítségével képezzük.

A kockajáték szimulálására a következő képletet használjuk:

INT (RND(1) * 6 + 1

Először 0—1 közötti véletlen számot megszorozzuk 6-tal, amely kiterjeszti a tartományt 0 és 6 közé (nagyobb, mint nulla, és kisebb, mint hat). Utána hozzáad egyet, és így kiterjeszti a tartományt 1 és 7 közé. Az INT függvény levágja az összes tizedes jegyet és így a kapott értékek egész számok 1 és 6 között.

Két kocka esetén összegezzük a fent kapott képletet.

Példa:

100 X = INT (RND(1) * 6) + INT (RND(1) * 6) + 2

és két kocka dobását szimulálja

100 X = INT (RND(1) * 1000) + 1

számok 1 és 1000 között

100 X = INT (RND(1) * 150) + 100

számok 100 és 249 között

SGN (X) (szignum, előjel)

Ezzel a függvénnyel állapíthatjuk meg az X előjelét. Az eredmény 1, ha X pozitív, 0, ha X = 0. és —1, ha X negatív.

SIN (X) (sinus)

Az X szög színuszát adja meg. A szöget radiánban kell megadni.

SOR (X) négyzetgyök

Megadja az X négyzetgyökét, ha X nagyobb vagy egyenlő 0. Negatív X esetén ILLEGAL QUANTITY ERROR hibajelzést kapunk.

TAN (X) (tangens)

Az eredmény az X szög tangense. X-et radiánban kell megadni . . .

USR (X)

Ha az USR függvényt használjuk, a program egy gépi kódú programra ugrat, amelynek kezdőcíme a 785 és 786 tárolóhelyen van elhelyezve. Az X paraméter átadódik a gépi kódú programnak, amely viszont a maga részéről egy másik értéket fog a BASIC programnak visszaadni.

b) String függvények

ASC (X$)

Megadja az X$ első karakterének ASCII kódját.

CHR$ (X)

Ez az ASC (X$) inverz függvénye, és megadja azt a karaktert, amelynek ASCII kódja X.

LEFT (X$, X)

A stringnek azt a részét adja meg, amely az X$ bal oldali X részét tartalmazza.

LEN (X$)

Megadja az X$-ban található karakterek számát (beleértve a space-eket és egyéb szimbólumokat is).

MID$ (X$, S, X)

Olyan stringet eredményez, amely XS-ból az S-edik karaktertől kezdődően X karaktert tartalmaz.

RIGHT$ (X$)

X$ jobb oldali X karakterét eredményezi.

STR$ (X)

Olyan stringet eredményez, amely az X PRINT útján kiadott változatával azonos.

VAL (X$)

Ez a függvény az X$-t számmá alakítja át, és lényegében a STR$ (X) inverz függvénye. A string vizsgálata a bal oldali első jeltől kezdve történik, mindaddig, amíg egy olyan jel nem fordul elő, amely nem felel meg valamely, a számábrázolásban megjeleníthető jelnek. Az így megállapított karaktersorozatot számmá alakítja át.

10 X = VAL ("123,456")	X = 123.456
10 X = VAL ("12A185")	X = 12
10 X = VAL ("RIUQk7"	X = 0
10 X = VAL ("–1.23.15.67")	X = –1.23

c) Egyéb függvények

FRE (X)

Ez a függvény a tároló szabad byte-jainak számát adja meg, az X értékétől függetlenül.

POS (X)

Ez a függvény a képernyő sorában azt a pozíciót adja meg (Ø–21), amelyen a következő PRINT utasítás végrehajtása következne. Az X bármilyen értékű lehet, nem használjuk.

SPC (X)

Ez a függvény PRINT utasításokban használatos X pozíció átugratására.

TAB (X)

A TAB függvény ugyancsak PRINT utasításokban használatos. Hatására a következő karakter az X-edik oszlopba kerül kinyomtatásra.

D) MELLÉKLET: A BASIC KULCSSZAVAK RÖVIDITÉSEI

Annak érdekében, hogy a programok és a közvetlen parancsok beadásakor időt takaríthassunk meg, a VIC legtöbb BASIC kulcsszava rövidíthető. A PRINT rövidítése egy kérdőjel. A többi kulcsszó rövidítései úgy képezhetők, hogy az első egy vagy két jelet beadjuk, majd a szóban levő következő jelet SHIFT-tel együtt adjuk be. Ha a rövidítéseket programokban használjuk, a LIST a kulcsszavakat teljes alakjukban írja ki. Ügyeljünk arra, hogy az egyes kulcsszavak rövidítései kezdő zárójelet is magukban foglalnak.

Parancs	Rövidítés	Képernyő
AND	A SHIFT N	A
NOT	N SHIFT O	N
CLOSE	CL SHIFT O	CL
CLR	C SHIFT L	C
CMD	C SHIFT M	C
CONT	C SHIFT O	C
DATA	D SHIFT A	D
DEF	D SHIFT E	D
DIM	D SHIFT I	D
END	E SHIFT N	E
FOR	F SHIFT O	F
GET	G SHIFT E	G
GOSUB	GO SHIFT S	GO
GOTO	G SHIFT O	G
INPUT#	I SHIFT N	I
LET	L SHIFT E	L
LIST	L SHIFT I	L
LOAD	L SHIFT O	L
NEXT	N SHIFT E	N
OPEN	O SHIFT P	O
POKE	P SHIFT O	P
PRINT	?	?
RND	R SHIFT N	R
SGN	S SHIFT G	S
SIN	S SHIFT I	S
SPC (	S SHIFT P	S
SQR	S SHIFT Q	S

Parancs	Rövidítés	Képernyő
PRINT#	P SHIFT R	P
READ	R SHIFT E	R
RESTORE	RE SHIFT S	RE
RETURN	RE SHIFT T	RE
RUN	R SHIFT U	R
SAVE	S SHIFT A	S
STEP	ST SHIFT E	ST
STOP	S SHIFT T	S
SYS	S SHIFT Y	S
THEN	T SHIFT H	T
VERIFY	V SHIFT E	V
WAIT	W SHIFT A	W
ABS	A SHIFT B	A
ASC	A SHIFT S	A
ATN	A SHIFT T	A
CHR$	C SHIFT H	C
EXP	E SHIFT X	E
FRE	F SHIFT R	F
LEFT$	LE SHIFT F	LE
MID$	M SHIFT I	M
PEEK	P SHIFT E	P
RIGHT$	R SHIFT I	R
STR$	ST SHIFT R	ST
TAB (	T SHIFT A	T
USR	U SHIFT S	U
VAL	V SHIFT A	V

E) MELLÉKLET

A képernyő alapszíne és a keret színe bármikor cserélhető a VIC-nél, programon belül vagy kívül a következő begépelésével:

POKE 36879, X

ahol X az alábbi táblázatból meghatározható.

POKE 36879,27 begépelésének a hatására a gép az alap színkombinációra áll be, ami fehér alap és kékeslila keret. Gépelje be: POKE 36879,8. Utána CTRL 2, és ekkor fehér betűi lesznek fekete képernyőn. Próbáljon ki más kombinációkat.

	HÁTTÉR							
ALAPSZÍN	BLK	WHT	RED	CYAN	PUR	GRN	BLU	YEL
Fekete	8	9	10	11	12	13	14	15
Fehér	24	25	26	27	28	29	30	31
Piros	40	41	42	43	44	45	46	47
Kékeslila (cián)	56	57	58	59	60	61	62	63
Bíbor	72	73	74	75	76	77	78	79
Zöld	88	89	90	91	92	93	94	95
Kék	104	105	106	107	108	109	110	111
Sárga	120	121	122	123	124	125	126	127
Narancs	136	137	138	139	140	141	142	143
Vil. narancs	152	153	154	155	156	157	158	159
Rózsaszín	168	169	170	171	172	173	174	175
Vil. kékeslila	184	185	186	187	188	189	190	191
Világoslila	200	201	202	203	204	205	206	207
Világoszöld	216	217	218	219	220	221	222	223
Világoskék	232	233	234	235	236	237	238	239
Vil. sárga	248	249	250	251	252	253	254	255

F) MELLÉKLET

Zenei hangok táblázata

Hangjegy	Érték	Hangjegy	Érték
C	135	G	215
C#	143	G#	217
D	147	A	219
D#	151	A#	221
E	159	H	223
F	163	C	225
F#	167	C#	227
G	175	D	228
G#	179	D#	229
A	183	E	231
A#	187	F	232
H	191	F#	233
C	195	G	235
C#	199	G#	236
D	201	A	237
D#	203	A#	238
E	207	H	239
F	209	C	240
F#	212	C#	241

Hanggenerátor parancs	X lehetséges értékei	Funkció
POKE 36878, X	0–15	Hangerő beállítás
POKE 36874, X	128–255	Hangjegy megszólal
POKE 36875, X	128–255	Hangjegy megszólal
POKE 36876, X	128–255	Hangjegy megszólal
POKE 36877, X	128–255	Zajgenerálás

G) MELLÉKLET: 20 PROGRAM A VIC 20-RA

Bemutatunk néhány mintaprogramot a későbbi programjaink hanggenerálásához. Begépelheti ezeket a VIC 20-ba magába, vagy egy programon belül. Természetesen ezzel nem merült ki a VIC 20 hanghatásainak lehetősége.

Az így bemutatott hangeffektusok a programon belül szünetet generálnak mindaddig, amíg be nem fejeződnek. Be lehet ezeket a hatásokat a programba úgy is építeni, hogy semmit nem állítanak le.

N felejtse el a sorszámok használatát, ha ezeket a rutinokat begépeli a számítógépbe. Most elhagytuk a sorszámokat, hogy az ön programjaiba történő begépeléskor ezek ne okozzanak zavart.

1. SKÁLÁK

```
POKE 36878,15
FOR L = 250 TO 200 STEP -2
POKE 36876,L
FOR M = 1 TO 100
NEXT M
NEXT L
FOR L = 205 TO 250 STEP 2
POKE 36876,L
FOR M = 1 TO 100
NEXT M
NEXT L
POKE 36876,0
POKE 36878,0
```

2. COMPUTER MÁNIA

```
POKE 36878,15
FOR L = 1 TO 100
POKE 36876, INT (RND(1)* 128) + 128
FOR M = 1 TO 10
NEXT M
NEXT L
POKE 36876,0
POKE 36878,0
```

3. ROBBANÁS

```
POKE 36877,220
FOR L = 15 TO STEP -1
POKE 36878,L
FOR M = 1 TO 300
NEXT M
NEXT L
POKE 36877,0
POKE 36878,0
```

4. BOMBÁK

```
POKE 36878,10
FOR L = 230 TO 128 STEP -1
POKE 36876,L
FOR M = 1 TO 20
NEXT M
NEXT L
POKE 36876,0
POKE 36877,200
FOR L = 15 TO 0 STEP - .05
POKE 36878, L
NEXT L
POKE 36877,0
```

5. ALARM

```
POKE 36878,15
FOR L = 1 TO 10
FOR M = 180 TO 235 STEP 2
POKE 36876,M
FOR N = 1 TO 10
NEXT N
NEXT M
POKE 36876,0
FOR M = 1 TO 100
NEXT M
NEXT L
POKE 36878,0
```

6. LÉZERSUGÁR

```
POKE 36878,15
FOR L = 1 TO 30
FOR M = 250 TO 240 STEP -1
POKE 36876,M
NEXT M
FOR M = 240 TO 250
POKE 36876,M
NEXT M
POKE 36876,0
NEXT L
POKE 36878,0
```

7. SZIRÉNA

```
POKE 36878,15
FOR L = 1 TO 10
POKE 36875,200
FOR M = 1 TO 500
NEXT M
POKE 36875,0
POKE 36876,200
FOR M = 1 TO 500
NEXT M
POKE 36876,0
NEXT L
POKE 36878,0
```

8. FOGLALT JELZÉS

```
POKE 36878,15
FOR L = 1 TO 15
POKE 36876,160
FOR M = 1 TO 400
NEXT M
POKE 36876,0
FOR M = 1 TO 400
NEXT M
NEXT L
POKE 36878,0
```

9. CSÖRÖG A TELEFON

```
POKE 36878,15
FOR L = 1 TO 5
FOR M = 1 TO 50
POKE 36876,220
FOR N = 1 TO 5
NEXT N
POKE 36876,0
NEXT M
FOR M = 1 TO 3000
NEXT M
NEXT L
POKE 36878,0
```

10. CSIRIPELŐ MADARAK

```
POKE 36878,15
FOR L = 1 TO 20
FOR M = 254 TO 240 + INT (RND(1)*10)STEP-1
POKE 36876,M
NEXT M
POKE 36876,0
FOR M = 0 TO INT(RND(1)*100) + 120
NEXT M
NEXT L
```

11. SZÉL

```
POKE 36878,15
POKE 36874,170
POKE 36877,240
FOR L = 1 TO 2000
NEXT L
POKE 36874,0
POKE 36877,0
POKE 36878,0
```

12. A TENGER HULLÁMAI

```
POKE 36877,180
FOR L = 1 TO 10
D = INT(RND(1)*5)*50 + 50
FOR M = 3 TO 15
POKE 36878,M
FOR N = 1 TO D
NEXT N
NEXT M
FOR M = 15 TO 3 STEP -1
POKE 36878,M
FOR N = 1 TO D
NEXT N
NEXT M
NEXT L
POKE 36878,0
POKE 36877,0
```

13. ELTŰNŐ UFO

```
POKE 36878,15
FOR L = 130 TO 254
POKE 36876,L
FOR M = 1 TO  40
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

14. LESZÁLLÓ UFO

```
POKE 36878,15
FOR L = 1 TO 20
FOR M = 220-L TO 160-L STEP —4
POKE 36876,M
NEXT M
FOR M = 160-L TO 220-L STEP 4
POKE 36876,M
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

15. LÖVÖLDÖZŐ UFO

```
POKE 36878,15
FOR L = 1 TO 15
FOR M = 200 TO 220 + L*2
POKE 36876,M
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

16. FÜTYÜLŐ FARKAS

```
POKE 36878,15
FOR L = 148 TO 220 STEP .7
POKE 36876,L
NEXT L
FOR L = 128 TO 200
POKE 36876,L
NEXT L
FOR L = 200 TO 128 STEP —1
POKE 36876,L
NEXT L
POKE 36878,0
POKE 36876,0
```

17. FUTÓ LÁBAK

```
POKE 36878,15
FOR L = 1 TO 10
POKE 36874,200
FOR M = 1 TO 10
NEXT M
POKE 36874,0
FOR M = 1 TO 100
NEXT M
NEXT L
POKE 36878,0
```

18. TIK-TAK

```
POKE 36878,15
FOR L = 1 TO 10
POKE 36875,200
FOR M = 1 TO 10
NEXT M
POKE 36875,0
FOR M = 1 TO 300
NEXT M
POKE 36874,200
FOR M = 1 TO 10
NEXT M
POKE 36874,0
FOR M = 1 TO 300
NEXT M
NEXT L
POKE 36878,0
```

19. NYÍLÓ AJTÓ

```
POKE 36878,15
B = 0
FOR L = 128 TO 255 STEP 11
POKE 36874,L
FOR M = 1 TO 10
NEXT M
B = B + 1
IF B = 3 THEN B = 0: POKE 36874,0
NEXT L
POKE 36874,0
POKE 36878,0
```

```
20. BIP
POKE 36878,15
POKE 36876,220
FOR L = 1 TO 5
NEXT L
POKE 36876,0
FOR L = TO 500
NEXT L
POKE 36876,200
FOR L = 1 TO 5
NEXT L
POKE 36876,0
FOR L = 1 TO 500
NEXT L
POKE 36878,0
```

H) MELLÉKLET: KÉPERNYŐ KÓDOK

A következő táblázat felsorolja a VIC karakterkészletében előforduló valamennyi jelet. Bemutatja, milyen számot kell a képernyő-tárolóban (7680—8185 cím) POKE-olni ahhoz, hogy egy bizonyos karaktert kapjunk. Fordítva pedig megmutatja, hogy a képernyőtárolóból PEEK-kel előhívott számnak melyik karakter felel meg. Kétféle karakterkészlet áll rendelkezésre, de egyszerre mindig csak az egyik. Ez azt jelenti, hogy nem tudunk az egyik karakterkészletből jeleket a képernyőre vinni, amíg ott a másik karakterkészlethez tartozó karakterek vannak. A két karakterkészlet közötti átváltás úgy történik, hogy a SHIFT és a Commodore gombot egyidejűleg nyomjuk meg.

Ez gyakorlatilag 2 bitet cserél meg a 36869 memória címen, ami azt jelenti, hogy a POKE 36869,240 a karakterkészletet nagybetűsre, a POKE 36869,242 pedig kisbetűsre váltja át. Ha különböző komolyabb rajzokat kíván készíteni, meglátja, hogy a képernyő pontjait könnyebb a POKE utasítással a memóriában kézben tartani (és törölni azokat a 32-vel, ami az üres karakter kódja, és ezt visszük be az adott memória címre), mint printelni a képernyőre a kurzor ellenőrző karakterek használatával. A táblázat valamennyi karaktere megjeleníthető REVERSE (fordított) ábrázolásban is (világos alapon sötét karakter), ha a megadott értékhez 128-at hozzáadunk.

Megjegyzés: lásd a képernyő memóriatérkép-függeléket

Amennyiben egy szívet kíván rajzoltatni a képernyőnek arra a pontjára, melyet a 7800 címen találunk, ebben az esetben keresse meg a karakter számát (jelen esetben a szívét) a táblázatban (a szívé 83), billentyűzze be a POKE utasítást, a memória címet (7800) és a szimbólum számát (83) a következőképp:

```
POKE 7800,83
```

Egy fehér szív jelenik meg a képernyő közepén. Ne tévesszen meg minket, hogy nem látjuk, ez azért van, mert a háttér is fehér. Próbálja meg a pozíció megváltoztatását, úgy, hogy nagyobb

számot használ, vagy válasszon más szimbólumot, melynek kódját megtalálja a táblázatban.

Amennyiben a kiírandó szimbólum színét akarja megváltoztatni, úgy vegye figyelembe azt a táblázatot is, mely az egyes címekhez tartozó színkódokat tartalmazza. Más szóval a különböző helyek különböző színű szimbólumai további POKE utasítást igényelnek.

Például ahhoz, hogy vörös szívet kapjunk, üssük be a következőt:

POKE 38520,2

A képernyő (háttér) esetén ez a szám eggyel kevesebb, mint a billentyűzeten található színbillentyűé.

Ez változtatja a 7800 címen található szimbólum színét pirosra. Ha más szimbóluma van ott, az vörös színű lesz. A rendelkezésre álló színek közül bármilyen karaktert tetszőleges színben képes kiírni a képernyőre. Ezeket a POKE utasításokat be lehet venni a programba, mivel — különösen a rajzok — készítésénél nagyon hatásosak, ezen kívül a PEEK utasítás használatával nagyon okos kis programok készíthetők (labdajátékok).

KÉPERNYŐ KÓDOK

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
@		0	U	u	21	*		42
A	a	1	V	v	22	+		43
B	b	2	W	w	23	,		44
C	c	3	X	x	24	—		45
D	d	4	Y	y	25			46
E	e	5	Z	z	26	/		47
F	f	6	[		27	ø		48
G	g	7	£		28	1		49
H	h	8	]		29	2		50
I	i	9	↑		30	3		51
J	j	10	←		31	4		52
K	k	11	SPACE		32	5		53
L	l	12	!		33	6		54
M	m	13	"		34	7		55
N	n	14	#		35	8		56
O	o	15	$		36	9		57
P	p	16	%		37			58
Q	q	17	&		38	;		59
R	r	18	'		39	<		60
S	s	19	(		40	=		61
T	t	20	)		41	>		62

SET 1	SET 2	POKE
?		63
		64
	A	65
	B	66
	C	67
	D	68
	E	69
	F	70
	G	71
	H	72
	I	73
	J	74
	K	75
	L	76
	M	77
	N	78
	O	79
	P	80
	Q	81
	R	82
	S	83

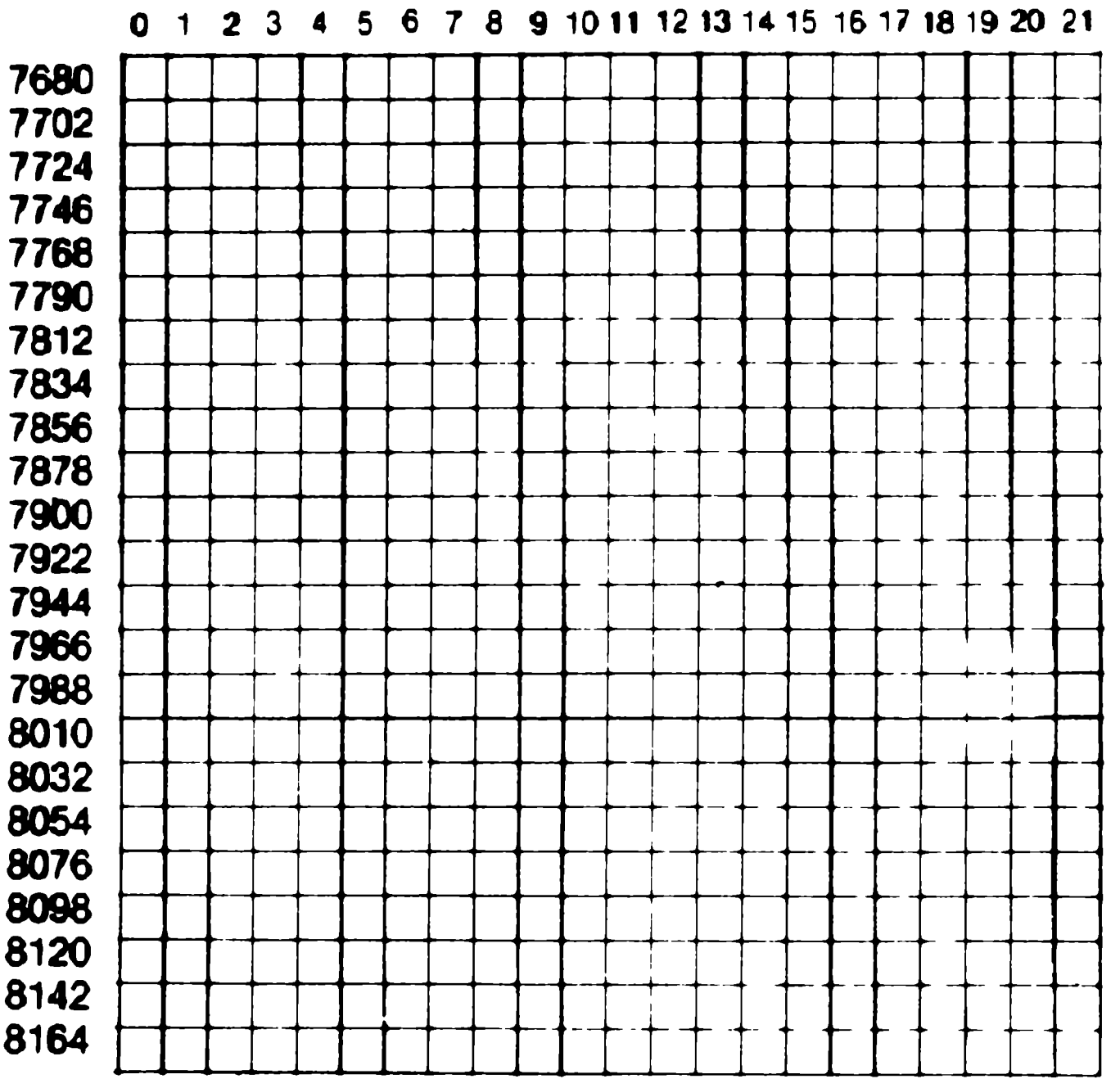
SET 1	SET 2	POKE
	T	84
	U	85
	V	86
	W	87
	X	88
	Y	89
	Z	90
		91
		92
		93
		94
		95
SPACE		96
		97
		98
		99
		100
		101
		102
		103
		104
		105

SET 1	SET 2	POKE
		106
		107
		108
		109
		110
		111
		112
		113
		114
		115
		116
		117
		118
		119
		120
		121
	✓	122
		123
		124
		125
		126
		127

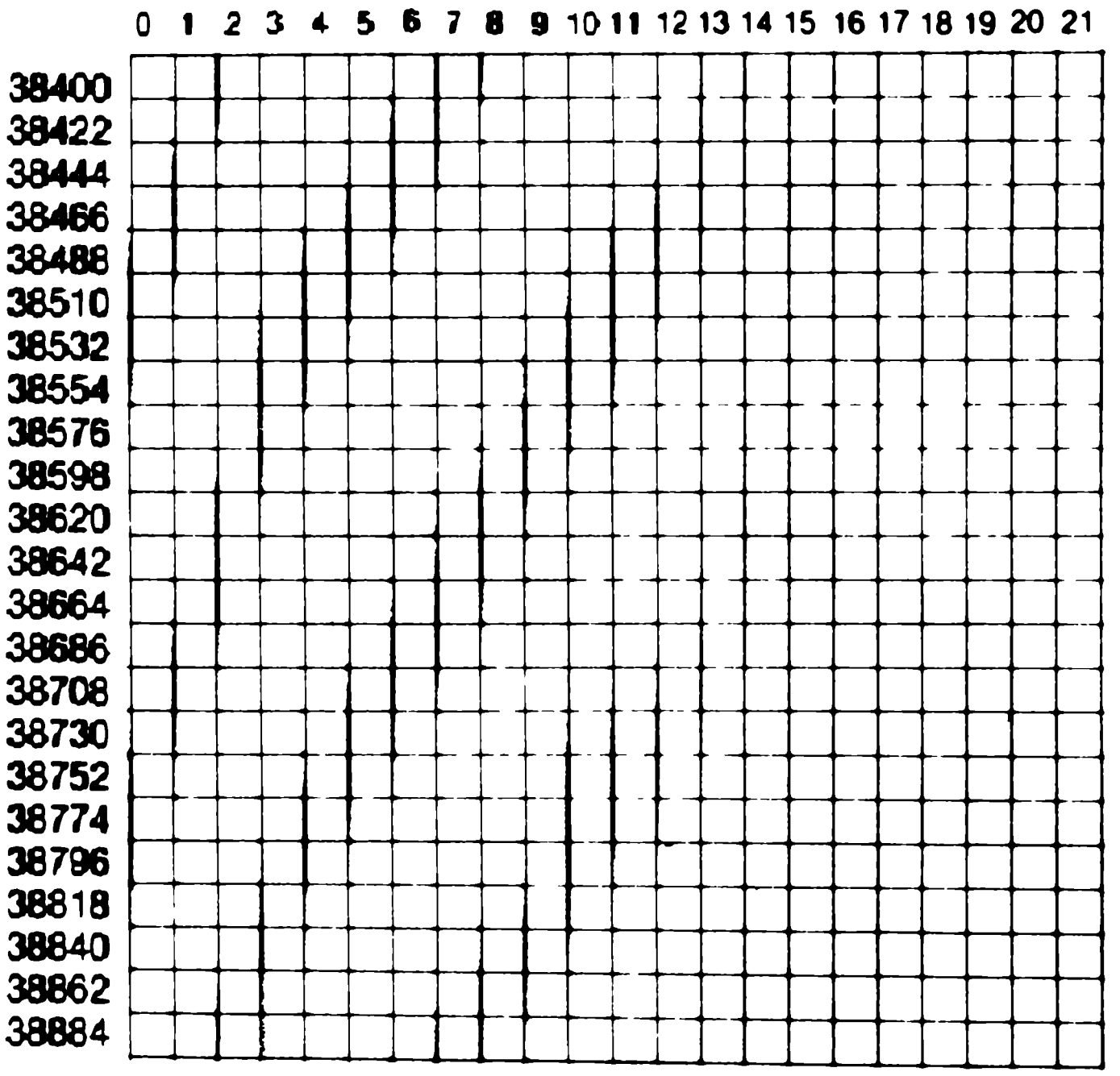
I) MELLÉKLET: KÉPERNYŐ MEMÓRIA TÉRKÉP

Ebben a mellékletben a képernyő bármelyik pozíciójának memóriacíme megtalálható. A rácshálózaton határozzuk meg a pozíciót, és adjuk össze a sorok és oszlopok számát. Amennyiben a grafikus labda karaktert akarja a képernyő közepére vinni, összeadja a 11. sor és 11. oszlop számát (7900 + 10) így 7910-et kap eredményül. Beviszi a labda kódját (81 ld. H) Melléklet) a 7910-es címre a POKE 7910,81 utasítással, és így egy fehér labda jelenik meg a képernyőn. A labda (vagy más karakter) színének a megváltoztatására keresse meg a megfelelő pozíciót a színkód-memóriatérképen, adja össze a sorok és oszlopok számait (3620 + 10 vagy 3630) a színkód meghatározására, és írja le második POKE utasításként. Például; ha színkódnak erre a címre a POKE 38630,3-t gépeli be, a labda cyan (kékeslila) színű lesz. Ne feledjük, POKE esetén a karakterek kódszáma eggyel kisebb, mint a színbillentyű száma — lásd: *Színkódok rövidített listája:*

Kód	Szín
0	fekete
1	fehér
2	piros
3	cyan
4	bíbor
5	zöld
6	kék
7	sárga



1.OLDAL: KÉPERNYŐ KARAKTER KÓDOK



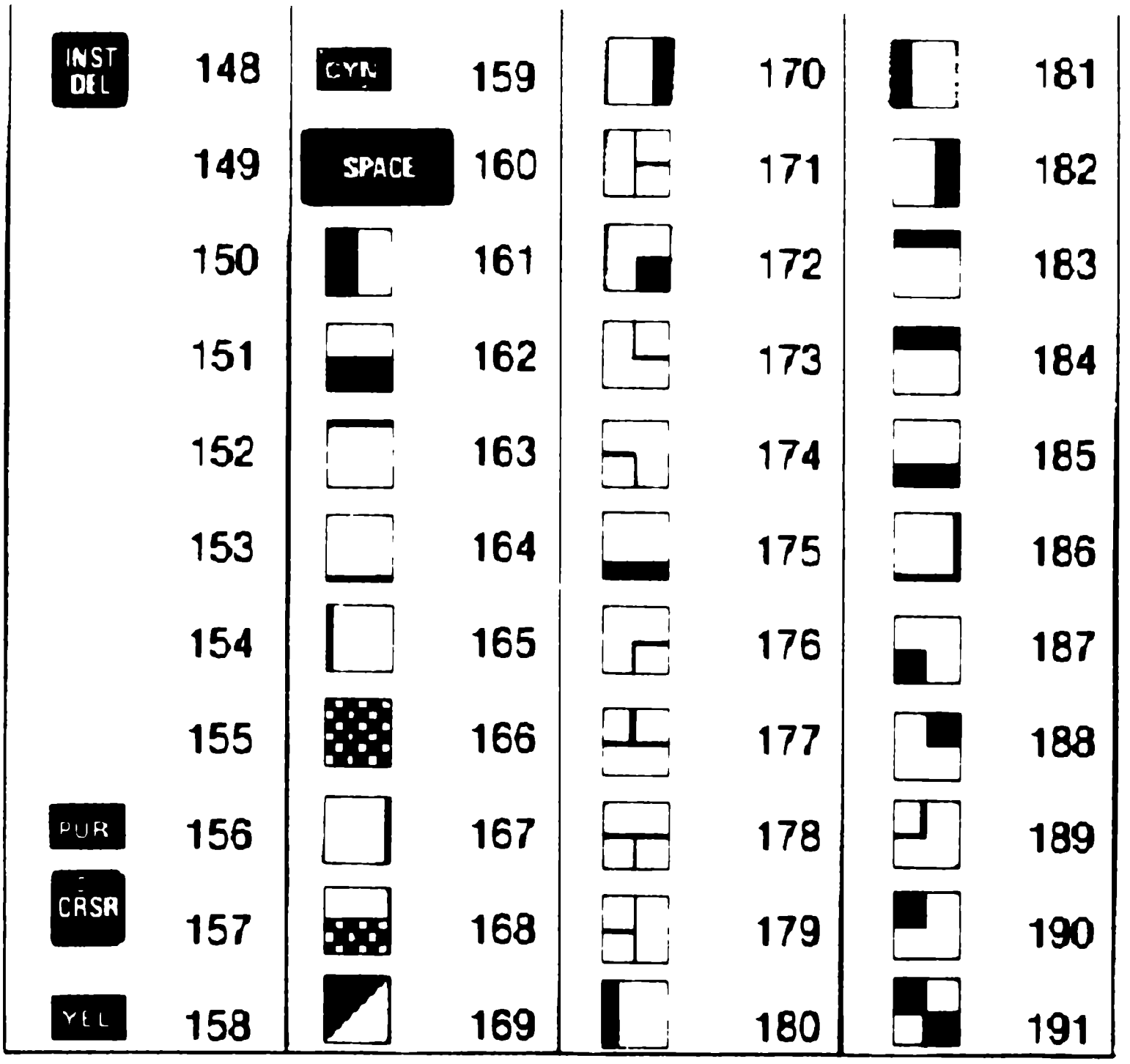
2. OLDAL: SZÍNKÓDOK MEMÓRIATÉRKÉPE

J) MELLÉKLET: ASCII ÉS CHR$ KÓDOK

Ez a függelék bemutatja azokat a karaktereket, melyek a PRINT CHR$ (X) hatására jelennek meg, az X összes lehetséges értéke mellett. Megmutatja azokat az értékeket, melyek a PRINT ASCII ("X") hatására jelennek meg, ahol C az ön által begépelt tetszőleges karakter. Célszerű felhasználása a GET utasítás hatására kapott karakter értékelése, kis/nagy betű átalakítás, karakter parancsok nyomtatása, melyeket nem lehet idézőjelbe tenni.

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
	0		16	SPACE	32	Ø	48
	1	CRSR	17	!	33	1	49
	2	RVS ON	18	"	34	2	50
	3	CLR HOME	19	#	35	3	51
	4	INST DEL	20	$	36	4	52
WHT	5		21	%	37	5	53
	6		22	&	38	6	54
	7		23	'	39	7	55
	8		24	(	40	8	56
	9		25	)	41	9	57
	10		26	*	42		58
	11		27	+	43	;	59
	12	RED	28	,	44	<	60
RETURN	13	CRSR	29	—	45	=	61
SWITCH TO LOWER CASE	14	GRN	30		46	>	62
	15	BLU	31	/	47	?	63

@	64	U	85		106		127
A	65	V	86		107		128
B	66	W	87		108		129
C	67	X	88		109		130
D	68	Y	89		110		131
E	69	Z	90		111		132
F	70	[	91		112	f1	133
G	71	£	92		113	f3	134
H	72	]	93		114	f5	135
I	73	↑	94		115	f7	136
J	74	←	95		111	f2	137
K	75		96		117	f4	138
L	76		97		118	f6	139
M	77		98		119	f8	140
N	78		99		120	SHIFT RETURN	141
O	79		100		121	SWITCH TO UPPER CASE	142
P	80		101		122		143
Q	81		102		123	BLK	144
R	82		103		124	CRSR	145
S	83		104		125	RVS OFF	146
T	84		105		126	CLR HOME	147



K) MELLÉKLET: MATEMATIKAI FÜGGVÉNYEK SZÁRMAZTATÁSA

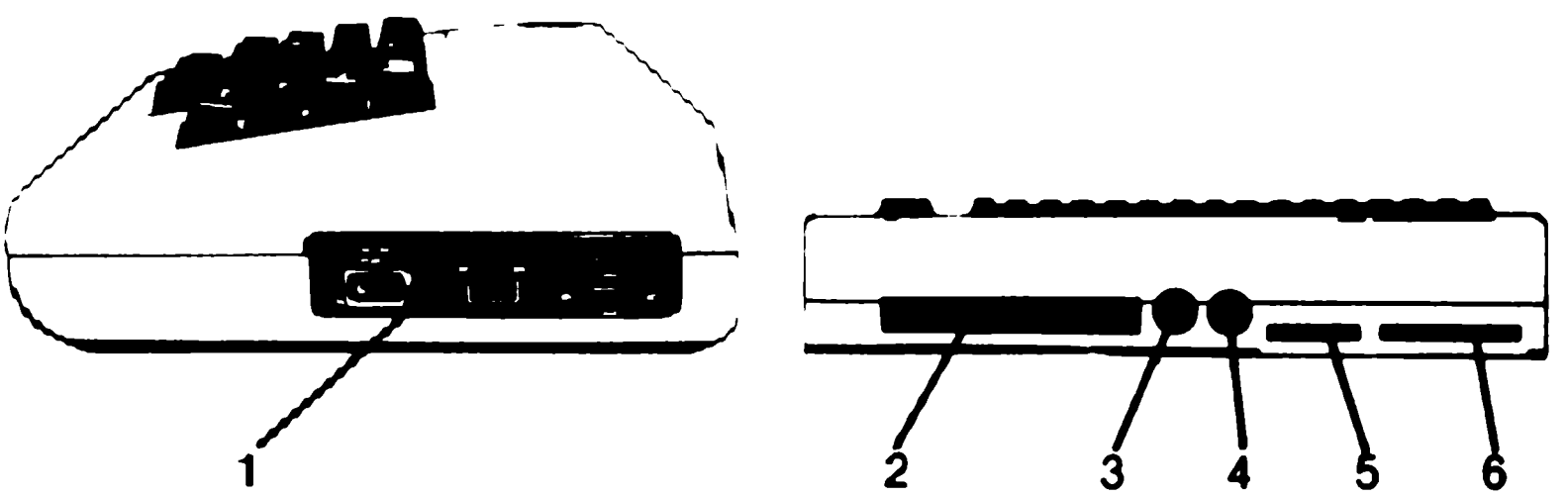
Azok a függvények, melyeket a BASIC nem ismer, a következőképpen számíthatók:

szekans	SEC (X) = 1/COS (X)
koszekans	CSC (X) = 1/SIN (X)
kotangens	CTG (X) = 1/TAN (X)
arkuszszinusz	ARCSIN (X) = ATN (X/SQR(−X * X + 1))
arkus koszinusz	ARCCOS (X) = −ATN (X/SQR(−X * X + 1)) + π/2
arkusz szekans	ARCSEC (X) = ATN (X/SQR(X * X−1))
arkusz koszekáns	ARCCSC (X) = ARCSEC (X) + (SGN (X)−1) * π/2
arkusz kotangens	ARCCTG (X) = ATN (X) + π/2
szinusz hiperbolikusz	SINH (X) = (EXP (X) − EXP (−X))/2
koszinusz hiperbolikusz	COSH (X) = (EXP (X) + EXP (−X))/2
tangens hiperbolikusz	TANH (X) = EXP (−X)/EXP(X) + EXP (−X)) * 2 + 1
kotangens hiperbolikusz	CTGH (X) = EXP (−X)/EXP(X)−EXP (−X)) * 2 + 1
szekans hiperbolikusz	SECH (X) = 2/(EXP (X) + EXP (−X))
koszekans hiperbolikusz	CSCH (X) = 2/(EXP (X) −EXP (−X))

arkuszszinusz hiperbolikusz	ARCSINH (X)	=	LOG (X + SQR (X * X + 1))
arkuszkoszinusz hiperbolikusz	ARCCOSH (X)	=	LOG (X + SQR (X * X − 1))
arkusztangens hiperbolikusz	ARCTANH (X)	=	LOG (1 + X) / (1−X))/2
arkuszkotangens hiperbolikusz	ARCCTGH (X)	=	LOG ((X + 1) / (X−1))/2
arkuszszekans hiperbolikusz	ARCSECH (X)	=	LOG (SGN (X) * SQR (X * X + 1) / X
koszekans hiperbolikusz	LOG ((SGN (X) * SQR (X * X + 1) /X		

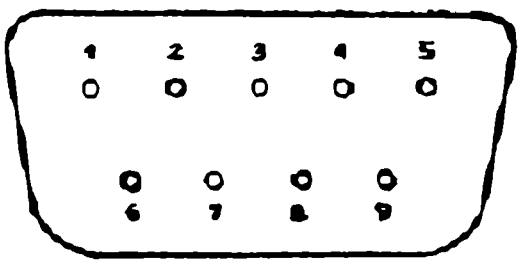
L) MELLÉKLET: INPUT/OUTPUT KÉSZÜLÉKEK LÁBKIOSZTÁSA

Itt láthatók a VIC I/O (bemenet/kimenet) csatlakoztatásai.



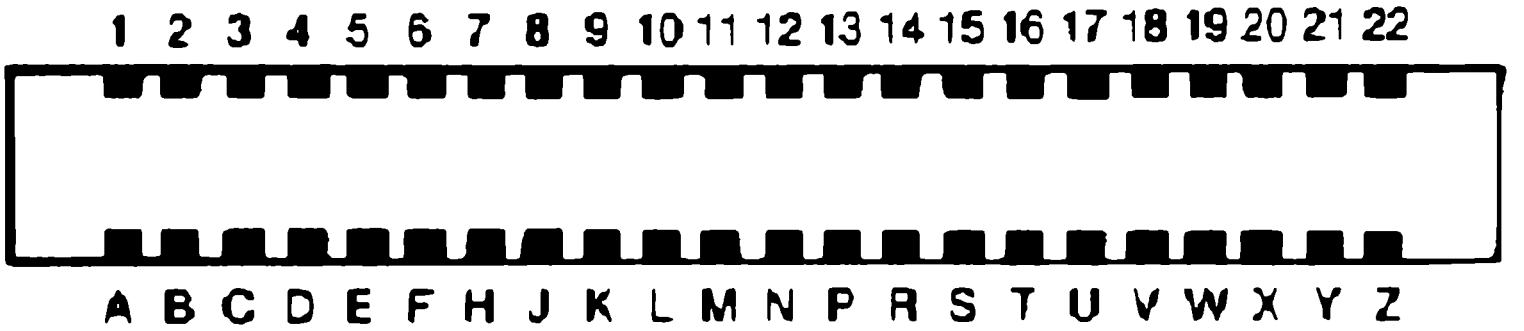
1. Játék I/O; 2. Memória bővítő; 3. Audió és Videó; 4. Soros I/O (diszk); 5. Kazetta egység; 6. Felhasználói pont (modem)

1. JÁTÉK I/O



LÁB#	TÍPUS	MEGJEGYZÉS
1	JOYØ	
2	JOY1	
3	JOY2	
4	JOY3	
5	POT Y	
6	LIGHT PEN	
7	+5V	MAX. 100mA
8	GND	
9	POT X	

2. MEMÓRIA BŐVÍTŐ



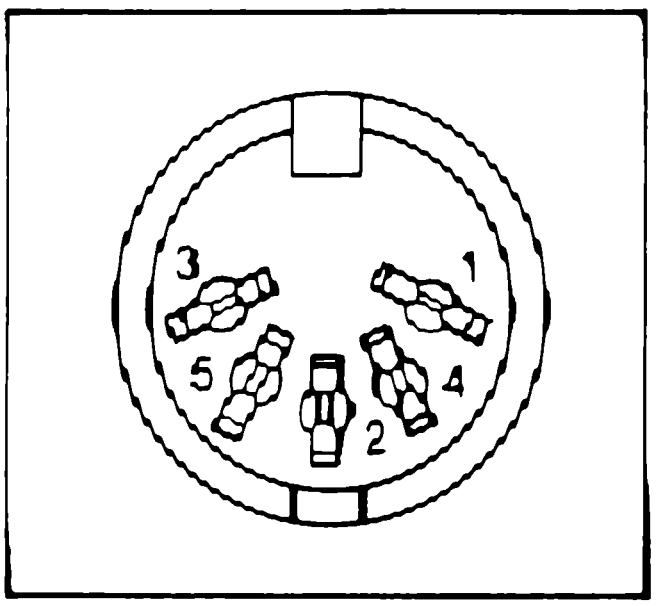
LÁB#	TÍPUS
1	GND
2	CDØ
3	CD1
4	CD2
5	CD3
6	CD4
7	CD5
8	CD6
9	CD7
1Ø	$\overline{\text{BLK1}}$
11	$\overline{\text{BLK2}}$

LÁB#	TÍPUS
12	$\overline{\text{BLK3}}$
13	$\overline{\text{BLK5}}$
14	$\overline{\text{RAM1}}$
15	$\overline{\text{RAM2}}$
16	$\overline{\text{RAM3}}$
17	VR/W
18	CR/W
19	$\overline{\text{IRQ}}$
2Ø	NC
21	+5V
22	GND

LÁB#	TÍPUS
A	GND
B	CAØ
C	CA1
D	CA2
E	CA3
F	CA4
H	CA5
J	CA6
K	CA7
L	CA8
M	CA9

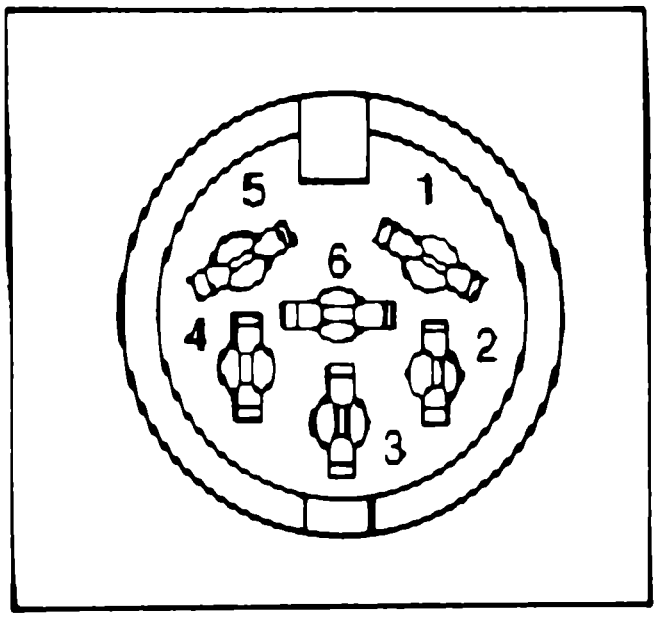
LÁB#	TÍPUS
N	CA1Ø
P	CA11
R	CA12
S	CA13
T	I/Ø2
U	I/Ø3
V	SØ2
W	$\overline{\text{NMI}}$
X	$\overline{\text{RESET}}$
Y	NC
Z	GND

3. AUDIÓ ÉS VIDEÓ



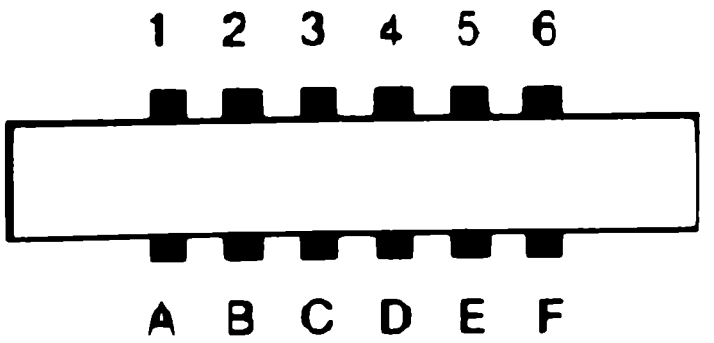
LÁB#	TÍPUS	MEGJEGYZÉS
1	+6V	10mA MAX
2	GND	
3	AUDIO	
4	VIDEO LOW	
5	VIDEO HIGH	

4. SOROS I/O /DISZK/



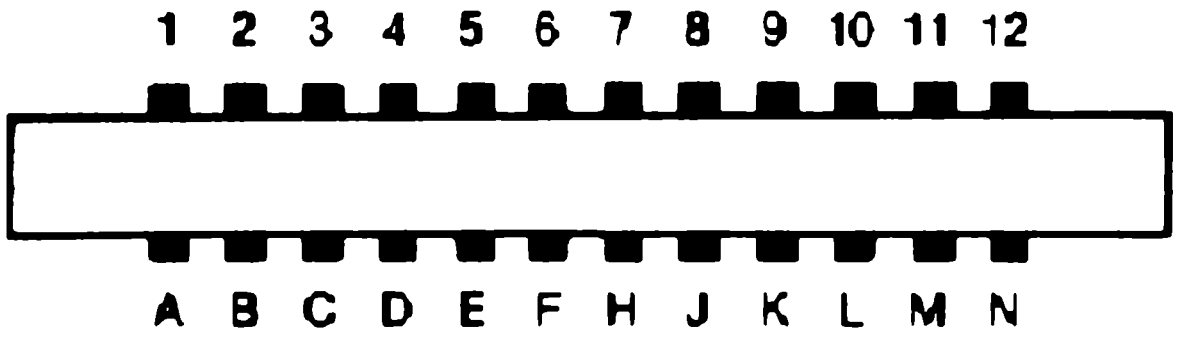
LÁB #	TÍPUS
1	SERIAL SRQ IN
2	GND
3	SERIAL ATN IN/OUT
4	SERIAL CLK IN/OUT
5	SERIAL DATA IN/OUT
6	NC

5. KAZETTA EGYSÉG



LÁB #	TÍPUS
A-1	GND
B-2	+5V
C-3	CASSETTE MOTOR
D-4	CASSETTE READ
E-5	CASSETTE WRITE
F-6	CASSETTE SWITCH

6. FELHASZNÁLÓI PONT /MODEM/



LÁB #	TÍPUS	MEGJEGYZÉS	LÁB #	TÍPUS	MEGJ.
1	GND		A	GND	
2	+5V	100mA MAX.	B	CB1	
3	RESET		C	PB0	
4	JOY0		D	PB1	
5	JOY1		E	PB2	
6	JOY2		F	PB3	
7	LIGHT PEN		H	PB4	
8	CASSETTE SWITCH		J	PB5	
9	SERIAL ATN IN		K	PB6	
10	+9V	100mA MAX.	L	PB7	
11	GND		M	CB2	
12	GND		N	GND	

M) MELLÉKLET: KIPRÓBÁLÁSRA AJÁNLOTT VIC PROGRAMOK

```
 0 REM "■■■■■■" GYILKOS BOLYGO
 1 C=7680+22*6: DIM B(12):LL=C:POKE 9*16↑3+15,8
 2 PRINT "■■*** GYILKOS BOLYGO ***":T=0
 3 PRINT " USS LE EGY BILLENTYUT"
 8 POKE 8179,160
 9 FOR A=38444 TO 38400+505:POKE A,1:NEXT
 10 :
 20 FOR A=1 TO 12: B(A)=160:NEXT
 21 IF W=0 THEN 25
 22 POKE 9*16↑3+14,U:U=U-2:IF U<=0 THEN W=0:POKE 9*16↑3+13,0
 25 F=1: REM METEOR TORLESE
 26 FOR E=0 TO 44 STEP 22
 27 FOR D=C+E TO C+3+E:POKE D,32: F=F+1: NEXT:NEXT
 28 C=C+1
 29 F=1: REM METEOR RAJZOLAS
 30 POKE 8179,160: FOR E=0 TO 44 STEP 22
 35 FOR D=C+E TO C+3+E:POKE D,B(F):F=F+1:NEXT:NEXT
 36 IF PEEK(8178)=160 THEN PRINT "■■■■■HOLD BAZIS LEROMBOLVA":GOTO 500
 40 GET A$:IF A$<>"" AND G=0 THEN G=1:S=7680+15+21*22
 50 IF G=0 THEN 80
 55 POKE S,32: S=S-22
 60 IF S<7746 THEN G=0: GOTO 21
 70 IFPEEK(S)=160THENPOKES,32:G=0:T=T+1:W=1:POKE9*16↑3+13,128:U=15:GOTO80
 71 IFPEEK(S-1)=160THENG=0:POKES-1,32:T=T+1:W=1:POKE9*16↑3+13,128:U=15:GOTO80
 75 POKE S,81
 80 F=1: REM METEOR VIZSGALAT
 81 IFT=12THENPRINT"■■METEOR LEROMBOLT*":FORRR=1TO2500:NEXT:LL=LL+44:C=LL:GOTO2
 82 FOR E=0 TO 44 STEP 22
 84 FOR D=C+E TO C+3+E: IF PEEK(D)=32 THEN B(F)=32
 86 F=F+1:NEXT:NEXT
 90 GOTO 21
500 POKE 9*16↑3+13,128+5
505 POKE 9*16↑3+14,5: FOR RR=1 TO 300:NEXT
510 FOR A=15 TO 0 STEP -1
511 POKE 9*16↑3+14,A
520 FOR RR=1 TO 500:NEXT
530 NEXT
540 FOR RR=1 TO 2000:NEXT:RUN
READY.
```

```
1 REM "*************"
2 REM "*************" RAKETA VEZERLES
10 VI=9*16↑3: OF=38400-7680:PRINT"♥"
11 FOR A=38400+22 TO 38400+22*23:POKE A,0:NEXT
15 C=7680+22*20+15
20 POKE VI+15,6+128+64+32+8
30 PRINT "■■* RAKETA VEZERLES *";
35 PRINT "USS LE EGY BILLENYUT"
40 PRINT "■■■■■■■■■■■■■■■■■■■■■■■■■■■■XXXXXXXXXXXXXXXXXXXXX";
50 PRINT "□■■■■■■■■■■■■■↑↑↑↑↑↑↑↑↑↑↑↑↑"
60 A=7680+22*22+15
70 GETA$:IFA$<>""ANDB=0THENB=1:POKEC+22,32:D=C:C=C-1:K=1:POKEVI+13,128+125:L=16
71 IF K=1 THEN L=L-1:POKE9*16↑3+14,L
72 IF L=0 THEN K=0:POKE 9*16↑3+13,0
75 IFC=8121 THEN PRINT"■■■■■■■NINCS TOBB RAKETA!!":POKE 9*16↑3+14,0
76 FOR AA=1 TO 3000:NEXT:RUN
80 IF B=0 THEN 110
85 POKE D,32:D=D-22
90 IF PEEK(D)=60 OR PEEK(D)=62 THEN GOSUB 500: GOTO 70
100 IF D<7680+88 THEN B=0:GOTO 110
105 POKE D,30
110 IF H=0 THEN 200
115 IF H=0 THEN 70
120 POKE F,32:F=F+E
125 IF F=I THEN H=0: GOTO 70
130 IF PEEK(F)=30 THEN GOSUB 500:GOTO 70
140 POKE F,G:GOTO 70
200 H=1
205 IFINT(RND(1)*2)=1THENE=-1:F=7702+(INT(RND(1)*10)+6)*22:I=F-22:G=60:GOTO115
210 E=1:F=7680+(INT(RND(1)*10)+6)*22:I=F+22:G=62:GOTO115
500 B=0:H=0
501 SC=SC+10:PRINT"■■■■■■EREDMENY="SC
502 POKE F+OF,4: POKE F+1+OF,4:POKE F-1+OF,4
503 POKE F+OF+22,4:POKE F-22+OF,4
510 POKEF,160:POKEF+1,160:POKEF-1,160:POKEF+22,160:POKEF-22,160
521 POKE 9*16↑3+13,128+35
522 FOR Y=16 TO 0 STEP -1
523 POKE 9*16↑3+14,Y
524 FOR P=1 TO 80:NEXT:NEXT
530 POKE F,32:POKE F+1,32:POKE F-1,32:POKE F+22,32: POKE F-22,32
531 POKE F+OF,0: POKE F+1+OF,0: POKE F-1+OF,0
532 POKE F+OF+22,0: POKE F-22+OF,0
533 POKE 9*16↑3+13,0
540 FOR GH=F TO F+22*16 STEP 22
544 II=PEEK(GH):POKE GH,G:FOR OO=1 TO 60:NEXT
546 POKE GH,II:NEXT
800 RETURN
999 GOTO 70
1000 POKE 9*16↑3+13,128+125
1001 FOR Y=16 TO 0 STEP -1
1005 POKE 9*16↑3+14,Y
1010 NEXT: POKE 9*16↑3+13,0
1020 RETURN
READY.
```

N) MELLÉKLET: HIBAJELZÉSEK

Ez a melléklet a VIC hibajelzéseinek teljes listáját tartalmazza hiba okainak leírásával.

BAD DATA

Egy file-ból string adatokat olvastunk, a program azonban numerikus adatokat várt.

BAD SUBSCRIPT

A program megpróbálta egy tömb egy elemét elérni, amely azonban a DIM utasításban megadott tartományon kívül esett.

CAN'T CONTINUE

A CONT utasítás nem működik, ha nem indítottunk el előtte RUN-nal egy programot, hiba lépett fel, vagy megváltoztattunk egy programsort.

DEVICE NOT PRESENT

Az elérni kívánt I/O egység nem volt elérhető OPEN, CLOSE, CND, PRINT, INPUT vagy GET esetén.

DIVISION BY ZERO

A nullával való osztás matematikailag n ncs definiálva és nem megengedett.

EXTRA IGNORED

INPUT-tal történt adatkérés után túl sok adat lett beadva. Csak az elsők lettek figyelembe véve.

FILE NOT FOUND

Ha szalagon kerestünk egy file-t, a gép END-OF-TAPE markert talált. Ha floppyn kerestünk egy file-t, nem létezik ilyen nevű file.

FILE NOT OPEN

A CMD, PRINT #, INPUT #, GET # útján elérni kívánt file-t először OPEN-nel meg kell nyitni.

FILE OPEN

Megpróbáltunk megnyitni egy file-t, és ehhez olyan logikai file-számot használtunk, amely már foglalt volt.

FORMULA TOO KOMPLEX

A string kifejezést legalább két részre kell osztani, hogy a rendszer fel tudja dolgozni.

ILLEGAL DIRECT

Az INPUT csak programon belül használható, közvetlen utasítás üzemmódban nem.

ILLEGAL QUANTITY

Az a szám, amelyet egy függvény vagy egy utasítás argumentumaként használtunk, a megengedett tartományon kívül esik.

LOAD

Probléma van a kazettán levő programmal.

NEXT WITHOUT FOR

Vagy helytelenül építettünk egymásba ciklusokat, vagy a NEXT-ben megadott változó nem egyezik a FOR-ban megadottal.

NOT INPUT FILE

Olyan file-ból próbáltunk meg INPUT # vagy GET # segítségével adatokat olvasni, amely csak outputra van értelmezve.

NOT OUTPUT FILE

Olyan file-ba próbáltunk PRINT segítségével íratni, amely csak olvasásra van megnyitva.

OUT OF DATA

READ utasítást próbáltunk végrehajtani, a DATA sorokban azonban nincs már több adat, amit még nem olvastunk volna.

OUT OF MEMORY

Nem áll már rendelkezésre több RAM terület a program vagy a változók számára. Ez a hiba akkor is felléphet, ha túl sok FOR ... NEXT ciklust vagy szubrutint építettünk egymásba, vagy túl sok zárójelet nyitottunk meg.

OVERFLOW

Egy számítás eredménye nagyobb, mint a megengedett legnagyobb szám (1.70141183E + 38).

REDIM'D ARRAY

Egy tömböt csak egyszer szabad DIMenzionálni. Ha mégis egy tömb-változóhoz fordul a program, mielőtt még az DIMenzionálva lett volna, a számítógép automatikusan elvégez egy DIM műveletet, amellyel a dimenziót tízre állítja be. Az ezután mégis következő DIM utasítás ezt a hibát okozza.

REDO FROM START

String jelek lettek beadva, holott az INPUT utasítás numerikus adat beadására várt. Ilyenkor nyugodtan adjuk be a helyes adatokat még egyszer, a program magától folytatódni fog.

RETURN WITHOUT GOSUB

A számítógép utasítást talált, előtte azonban nem volt GOSUB utasítás.

STRING TOO LONG

Egy string legfeljebb 255 jelet tartalmazhat.

SYNTAX

A VIC nem tud felismerni egy utasítást. Elfelejtettünk egy zárójelet, vagy éppen túl sokat adtunk meg, hibásan adtunk be egy kulcsszót stb.

TYPE MISMATCH

Ez a hiba akkor áll elő, ha számot használtunk egy string helyett és megfordítva.

UNDEF'D FUNCTION

Olyan, általunk definiált függvényre hivatkoztunk, amelyet még nem adtunk meg DEFFN utasításban, vagy a hozzá tartozó DEF FN utasításon a program még nem haladt át.

UNDEF'D STATEMENT

GOTO, GOSUB vagy RUN utasítással nem létező címkére próbáltunk ugratni.

VERIFY

A szalagon vagy floppyn található program nem egyezik a tárolóban levő programmal.

235 Ft